**LAMPIRAN 1**

**LISTING PROGRAM**

pom.xml

|  |
| --- |
| <project xmlns=**"http://maven.apache.org/POM/4.0.0"** xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**  xsi:schemaLocation=**"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd"**>  <modelVersion>**4.0.0**</modelVersion>  <groupId>**id.ac.gunadarma.tugasku**</groupId>  <artifactId>**TugasKu**</artifactId>  <packaging>**war**</packaging>  <version>**1.0-SNAPSHOT**</version>  <name>**Tugasku Maven Webapp**</name>  <properties>  <spring.version>**3.0.5.RELEASE**</spring.version>  </properties>  <dependencies>  <!-- Spring 3 dependencies -->  <dependency>  <groupId>**org.springframework**</groupId>  <artifactId>**spring-core**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**org.springframework**</groupId>  <artifactId>**spring-web**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**org.springframework**</groupId>  <artifactId>**spring-webmvc**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <!-- Spring Security -->  <dependency>  <groupId>**org.springframework.security**</groupId>  <artifactId>**spring-security-core**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**org.springframework.security**</groupId>  <artifactId>**spring-security-web**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**org.springframework**</groupId>  <artifactId>**spring-orm**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**org.springframework.security**</groupId>  <artifactId>**spring-security-config**</artifactId>  <version>**${spring.version}**</version>  </dependency>  <dependency>  <groupId>**cglib**</groupId>  <artifactId>**cglib**</artifactId>  <version>**2.2.2**</version>  </dependency>  <!-- jstl -->  <dependency>  <groupId>**javax.servlet**</groupId>  <artifactId>**jstl**</artifactId>  <version>**1.2**</version>  </dependency>  <dependency>  <groupId>**org.hibernate**</groupId>  <artifactId>**hibernate-core**</artifactId>  <version>**4.1.7.Final**</version>  </dependency>  <dependency>  <groupId>**postgresql**</groupId>  <artifactId>**postgresql**</artifactId>  <version>**8.3-606.jdbc4**</version>  <scope>**compile**</scope>  </dependency>  <dependency>  <groupId>**org.hibernate**</groupId>  <artifactId>**hibernate-c3p0**</artifactId>  <version>**4.1.7.Final**</version>  </dependency>  <dependency>  <groupId>**org.hibernate**</groupId>  <artifactId>**hibernate-entitymanager**</artifactId>  <version>**4.1.7.Final**</version>  </dependency>  <dependency>  <groupId>**log4j**</groupId>  <artifactId>**log4j**</artifactId>  <version>**1.2.17**</version>  </dependency>  <dependency>  <groupId>**javax.servlet**</groupId>  <artifactId>**servlet-api**</artifactId>  <version>**2.4**</version>  <scope>**provided**</scope>  </dependency>  <dependency>  <groupId>**org.freemarker**</groupId>  <artifactId>**freemarker**</artifactId>  <version>**2.3.19**</version>  </dependency>  <dependency>  <groupId>**org.codehaus.jackson**</groupId>  <artifactId>**jackson-mapper-asl**</artifactId>  <version>**1.9.10**</version>  </dependency>  <dependency>  <groupId>**mysql**</groupId>  <artifactId>**mysql-connector-java**</artifactId>  <version>**5.1.17**</version>  <type>**jar**</type>  <scope>**compile**</scope>  </dependency>  <dependency>  <groupId>**postgresql**</groupId>  <artifactId>**postgresql**</artifactId>  <version>**9.1-901.jdbc4**</version>  <scope>**compile**</scope>  </dependency>  <dependency>  <groupId>**commons-fileupload**</groupId>  <artifactId>**commons-fileupload**</artifactId>  <version>**1.2.2**</version>  </dependency>  <dependency>  <groupId>**commons-fileupload**</groupId>  <artifactId>**commons-fileupload**</artifactId>  <version>**1.2.2**</version>  </dependency>  <dependency>  <groupId>**commons-io**</groupId>  <artifactId>**commons-io**</artifactId>  <version>**2.4**</version>  </dependency>  </dependencies>  <build>  <finalName>**tugasku**</finalName>  <plugins>  <plugin>  <artifactId>**maven-compiler-plugin**</artifactId>  <configuration>  <source>**1.6**</source>  <target>**1.6**</target>  </configuration>  </plugin>  <plugin>  <groupId>**org.mortbay.jetty**</groupId>  <artifactId>**maven-jetty-plugin**</artifactId>  <configuration>  <contextPath>**/**</contextPath>  </configuration>  </plugin>  <plugin>  <groupId>**org.apache.maven.plugins**</groupId>  <artifactId>**maven-dependency-plugin**</artifactId>  <version>**2.3**</version>  <executions>  <execution>  <phase>**package**</phase>  <goals>  <goal>**copy**</goal>  </goals>  <configuration>  <artifactItems>  <artifactItem>  <groupId>**com.github.jsimone**</groupId>  <artifactId>**webapp-runner**</artifactId>  <version>**7.0.40.0**</version>  <destFileName>**webapp-runner.jar**</destFileName>  </artifactItem>  </artifactItems>  </configuration>  </execution>  </executions>  </plugin>  </plugins>  </build>  </project> |

LoginController.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**controller**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**EntityListWrapper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**security**.**UserService**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**service**.**TaskService**;**  **import** java**.**util**.**UUID**;**  **import** org**.**apache**.**commons**.**logging**.**Log**;**  **import** org**.**apache**.**commons**.**logging**.**LogFactory**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**stereotype**.**Controller**;**  **import** org**.**springframework**.**ui**.**ModelMap**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**ModelAttribute**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestMapping**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestMethod**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestParam**;**  **import** org**.**springframework**.**web**.**servlet**.**ModelAndView**;**  @Controller  public class LoginController **{**    private static Log LOG **=** LogFactory**.**getLog**(**LoginController**.**class**);**  @Autowired  private UserService userService**;**    @Autowired  private TaskService taskService**;**    @RequestMapping**(**value **=** "/"**,** method **=** RequestMethod**.**GET**)**  public ModelAndView index**()** **{**  **return** **new** ModelAndView**(**"redirect:/login"**);**  **}**    @RequestMapping**(**value **=** "/registerweb"**,** method **=** RequestMethod**.**GET**)**  public String registerGet**()** **{**  **return** "register"**;**  **}**    @RequestMapping**(**value **=** "/registerweb"**,** method **=** RequestMethod**.**POST**)**  public ModelAndView registerPost**(**@ModelAttribute User user**)** **{**  LOG**.**info**(**"Register User "**+**user**.**getUsername**());**  String generatedToken **=** UUID**.**randomUUID**().**toString**().**replace**(**"-"**,** ""**);**  user**.**setToken**(**generatedToken**);**  userService**.**addUser**(**user**);**  **return** **new** ModelAndView**(**"redirect:/login?success=true"**);**  **}**    @RequestMapping**(**value **=** "/login"**,** method **=** RequestMethod**.**GET**)**  public String login**(**ModelMap map**,** @RequestParam**(**value **=** "success"**,** defaultValue **=** ""**)** String success**)** **{**  map**.**addAttribute**(**"success"**,** success**);**  **return** "login"**;**  **}**  @RequestMapping**(**value **=** "/index"**,** method **=** RequestMethod**.**GET**)**  public String taskGet**(**ModelMap map**,**  @RequestParam**(**value **=** "success"**,** defaultValue **=** ""**)** String success**,**  @RequestParam**(**value **=** "page"**,** defaultValue **=** "1"**)** int page**)** **{**  int max **=** 5**;**  EntityListWrapper**<**Task**>** tasks **=** taskService**.**find**(**""**,** max**,** page **-** 1**);**  map**.**addAttribute**(**"tasks"**,** tasks**);**  map**.**addAttribute**(**"max"**,** max**);**  map**.**addAttribute**(**"page"**,** page**);**  **return** "index"**;**  **}**    @RequestMapping**(**value **=** "/loginfailed"**,** method **=** RequestMethod**.**GET**)**  public String loginerror**(**ModelMap model**)** **{**  model**.**addAttribute**(**"success"**,** "false"**);**  model**.**addAttribute**(**"error"**,** "true"**);**  **return** "login"**;**  **}**    @RequestMapping**(**value **=** "/logout"**,** method **=** RequestMethod**.**GET**)**  public ModelAndView logout**(**ModelMap model**)** **{**  **return** **new** ModelAndView**(**"redirect:/login"**);**  **}**    **}** |

Task.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**model**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**UUID**;**  **import** javax**.**persistence**.**CascadeType**;**  **import** javax**.**persistence**.**Column**;**  **import** javax**.**persistence**.**Entity**;**  **import** javax**.**persistence**.**Id**;**  **import** javax**.**persistence**.**ManyToOne**;**  **import** javax**.**persistence**.**PrePersist**;**  **import** javax**.**persistence**.**PreUpdate**;**  **import** javax**.**persistence**.**Table**;**  @Entity  @Table**(**name **=** "task"**)**  public class Task **{**  private String id **=** UUID**.**randomUUID**().**toString**();**  private Date created**;**  private String content**;**  private User user**;**  private Date deadline**;**  private boolean done**;**  private boolean deleted **=** **false;**  @Column  public String getContent**()** **{**  **return** content**;**  **}**  public void setContent**(**String content**)** **{**  **this.**content **=** content**;**  **}**  @ManyToOne**(**cascade **=** CascadeType**.**ALL**)**  public User getUser**()** **{**  **return** user**;**  **}**  public void setUser**(**User user**)** **{**  **this.**user **=** user**;**  **}**  @Column  public Date getDeadline**()** **{**  **return** deadline**;**  **}**  public void setDeadline**(**Date deadline**)** **{**  **this.**deadline **=** deadline**;**  **}**  @Column  public boolean isDone**()** **{**  **return** done**;**  **}**  public void setDone**(**boolean done**)** **{**  **this.**done **=** done**;**  **}**  @Override  public String toString**()** **{**  **return** "Task [content=" **+** content **+** ", user=" **+** user **+** ", deadline="  **+** deadline**.**toString**()** **+** ", done=" **+** done **+** "]"**;**  **}**  @Column  public boolean isDeleted**()** **{**  **return** deleted**;**  **}**  public void setDeleted**(**boolean deleted**)** **{**  **this.**deleted **=** deleted**;**  **}**  @Id  public String getId**()** **{**  **return** id**;**  **}**  public void setId**(**String id**)** **{**  **this.**id **=** id**;**  **}**  @Column  public Date getCreated**()** **{**  **return** created**;**  **}**  public void setCreated**(**Date created**)** **{**  **this.**created **=** created**;**  **}**  @PrePersist  @PreUpdate  protected void onCreate**()** **{**  created **=** **new** Date**();**  **}**  **}** |

User.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**model**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**HashSet**;**  **import** java**.**util**.**Set**;**  **import** javax**.**persistence**.**Access**;**  **import** javax**.**persistence**.**AccessType**;**  **import** javax**.**persistence**.**CascadeType**;**  **import** javax**.**persistence**.**Column**;**  **import** javax**.**persistence**.**Entity**;**  **import** javax**.**persistence**.**FetchType**;**  **import** javax**.**persistence**.**GeneratedValue**;**  **import** javax**.**persistence**.**Id**;**  **import** javax**.**persistence**.**JoinColumn**;**  **import** javax**.**persistence**.**JoinTable**;**  **import** javax**.**persistence**.**ManyToMany**;**  **import** javax**.**persistence**.**PrePersist**;**  **import** javax**.**persistence**.**PreUpdate**;**  **import** javax**.**persistence**.**Table**;**  **import** org**.**hibernate**.**annotations**.**GenericGenerator**;**  @Entity  @Table**(**name **=** "etask\_user"**)**  @Access**(**AccessType**.**FIELD**)**  public class User **{**    @Id  @GeneratedValue**(**generator **=** "system-uuid"**)**  @GenericGenerator**(**name **=** "system-uuid"**,** strategy **=** "uuid"**)**  private String id**;**    @Column  private Date created**;**    @Column**(**name **=** "first\_name"**)**  private String fname**;**  @Column**(**name **=** "last\_name"**)**  private String lname**;**  @Column**(**name **=** "username"**,** unique **=** **true)**  private String username**;**  @Column**(**name **=** "password"**)**  private String password**;**  @Column**(**name **=** "email"**,** unique**=true)**  private String email**;**  @Column**(**name **=** "token"**)**  private String token**;**  @ManyToMany**(**cascade **=** CascadeType**.**ALL**,** fetch **=** FetchType**.**LAZY**,** targetEntity **=** Role**.**class**)**  @JoinTable**(**name **=** "user\_role"**,** joinColumns **=** **{** @JoinColumn**(**name **=** "USER\_ID"**)** **},** inverseJoinColumns **=** **{** @JoinColumn**(**name **=** "ROLE\_ID"**)** **})**  private Set**<**Role**>** userSecurityRoleEntity **=** **new** HashSet**<**Role**>(**0**);**  public String getFname**()** **{**  **return** fname**;**  **}**  public void setFname**(**String fname**)** **{**  **this.**fname **=** fname**;**  **}**  public String getLname**()** **{**  **return** lname**;**  **}**  public void setLname**(**String lname**)** **{**  **this.**lname **=** lname**;**  **}**  public String getUsername**()** **{**  **return** username**;**  **}**  public void setUsername**(**String username**)** **{**  **this.**username **=** username**;**  **}**  public String getPassword**()** **{**  **return** password**;**  **}**  public void setPassword**(**String password**)** **{**  **this.**password **=** password**;**  **}**  public String getEmail**()** **{**  **return** email**;**  **}**  public void setEmail**(**String email**)** **{**  **this.**email **=** email**;**  **}**  public Set**<**Role**>** getUserSecurityRoleEntity**()** **{**  **return** userSecurityRoleEntity**;**  **}**  public void setUserSecurityRoleEntity**(**Set**<**Role**>** userSecurityRoleEntity**)** **{**  **this.**userSecurityRoleEntity **=** userSecurityRoleEntity**;**  **}**  public String getToken**()** **{**  **return** token**;**  **}**  public void setToken**(**String token**)** **{**  **this.**token **=** token**;**  **}**    public String getId**()** **{**  **return** id**;**  **}**  public void setId**(**String id**)** **{**  **this.**id **=** id**;**  **}**    public Date getCreated**()** **{**  **return** created**;**  **}**  public void setCreated**(**Date created**)** **{**  **this.**created **=** created**;**  **}**  @PrePersist  @PreUpdate  protected void onCreate**()** **{**  created **=** **new** Date**();**  **}**  **}** |

TokenInterceptor.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**security**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** javax**.**persistence**.**NoResultException**;**  **import** javax**.**servlet**.**http**.**HttpServletRequest**;**  **import** javax**.**servlet**.**http**.**HttpServletResponse**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**web**.**context**.**request**.**RequestAttributes**;**  **import** org**.**springframework**.**web**.**context**.**request**.**RequestContextHolder**;**  **import** org**.**springframework**.**web**.**servlet**.**handler**.**HandlerInterceptorAdapter**;**  public class TokenInterceptor **extends** HandlerInterceptorAdapter **{**    @Autowired  private UserService userService**;**    @Override  public boolean preHandle**(**HttpServletRequest request**,**  HttpServletResponse response**,** Object handler**)** **throws** Exception **{**  String accessToken **=** request**.**getHeader**(**"Authorization"**);**  User user **=** **null;**  **try** **{**  user **=** userService**.**getUserByToken**(**accessToken**);**  **}** **catch** **(**NoResultException e**)** **{**  e**.**printStackTrace**();**  **}**  RequestContextHolder**.**currentRequestAttributes**().**setAttribute**(**"user"**,** user**,** RequestAttributes**.**SCOPE\_SESSION**);**  **if** **(**request**.**getRequestURI**().**startsWith**(**"/api/"**)** **&&** user **==** **null)** **{**  response**.**sendError**(**javax**.**servlet**.**http**.**HttpServletResponse**.**SC\_UNAUTHORIZED**);**  **return** **false;**  **}**  **return** **true;**  **}**    public static User currentUser**()** **{**  **return** **(**User**)** RequestContextHolder**.**currentRequestAttributes**().**getAttribute**(**"user"**,** RequestAttributes**.**SCOPE\_SESSION**);**  **}**  **}** |

UserDetailsServiceImpl.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**security**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Role**;**  **import** java**.**util**.**Collection**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**dao**.**DataAccessException**;**  **import** org**.**springframework**.**security**.**core**.**userdetails**.**UserDetails**;**  **import** org**.**springframework**.**security**.**core**.**userdetails**.**UserDetailsService**;**  **import** org**.**springframework**.**security**.**core**.**userdetails**.**UsernameNotFoundException**;**  **import** org**.**springframework**.**stereotype**.**Service**;**  **import** org**.**springframework**.**transaction**.**annotation**.**Transactional**;**  @Service**(**"userDetailsService"**)**  public class UserDetailsServiceImpl **implements** UserDetailsService **{**  @Autowired  private UserService userService**;**  @Transactional**(**readOnly **=** **true)**  public UserDetails loadUserByUsername**(**String username**)**  **throws** UsernameNotFoundException**,** DataAccessException **{**  User userEntity **=** userService**.**findByName**(**username**);**  **if** **(**userEntity **==** **null)**  **throw** **new** UsernameNotFoundException**(**"user not found"**);**  String password **=** userEntity**.**getPassword**();**  Collection**<**Role**>** authorities **=** userEntity  **.**getUserSecurityRoleEntity**();**  **return** **new** org**.**springframework**.**security**.**core**.**userdetails**.**User**(**username**,**  password**,** **true,** **true,** **true,** **true,** authorities**);**  **}**  **}** |

UserService.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**security**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Role**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**EntityListWrapper**;**  **import** java**.**util**.**List**;**  public interface UserService **{**    public void removeUser**(**Integer id**);**  public List**<**User**>** listUser**();**  public void addUser**(**User user**);**  public void updateUser**(**User user**);**  public User getUserByID**(**Integer id**);**    public User getUser**(**String username**,** String password**);**  public User findByName**(**String username**);**  public User updateToken**(**User user**,** String generatedToken**);**  public User getUserByToken**(**String accessToken**);**    public Role getRoleById**(**String id**);**    public Role getRoleByAuth**(**String auth**);**    public EntityListWrapper**<**User**>** all**(**int max**,** int page**);**    **}** |

UserServiceImpl.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**security**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Role**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**EntityListWrapper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**PagingUtils**;**  **import** java**.**util**.**List**;**  **import** javax**.**persistence**.**EntityManager**;**  **import** javax**.**persistence**.**NoResultException**;**  **import** javax**.**persistence**.**PersistenceContext**;**  **import** javax**.**persistence**.**Query**;**  **import** org**.**springframework**.**stereotype**.**Repository**;**  **import** org**.**springframework**.**transaction**.**annotation**.**Transactional**;**  @Repository  @Transactional**(**readOnly **=** **true)**  public class UserServiceImpl **implements** UserService **{**  @PersistenceContext  private EntityManager entityManager**;**  @Transactional**(**readOnly**=false)**  public void addUser**(**User user**)** **{**  **try** **{**  entityManager**.**persist**(**user**);**  **}** **catch** **(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**  **}**  public User findByName**(**String username**)** **{**  Query query **=** entityManager  **.**createQuery**(**  "SELECT u FROM User u WHERE u.username = :username"**,**  User**.**class**);**  query**.**setParameter**(**"username"**,** username**);**  **try** **{**  **return** **(**User**)** query**.**getSingleResult**();**  **}** **catch** **(**NoResultException e**)** **{**  **return** **null;**  **}**  **}**  public User getUserByID**(**Integer id**)** **{**  Query query **=** entityManager**.**createQuery**(**  "SELECT u FROM User u WHERE id = :id"**,** User**.**class**);**  query**.**setParameter**(**"id"**,** id**);**  **try** **{**  **return** **(**User**)** query**.**getSingleResult**();**  **}** **catch** **(**NoResultException e**)** **{**  **return** **null;**  **}**  **}**  @Transactional  public void updateUser**(**User user**)** **{**  **try** **{**  entityManager**.**persist**(**user**);**  **}** **catch** **(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**  **}**  @SuppressWarnings**(**"unchecked"**)**  public List**<**User**>** listUser**()** **{**  **return** entityManager**.**createQuery**(**"SELECT u FROM User u"**)**  **.**getResultList**();**  **}**  @Transactional  public void removeUser**(**Integer id**)** **{**  User user **=** **(**User**)** entityManager**.**find**(**User**.**class**,** id**);**  **if** **(null** **!=** user**)** **{**  entityManager**.**remove**(**user**);**  **}**  **}**  @Override  public User getUser**(**String username**,** String password**)** **{**  Query query **=** entityManager**.**createQuery**(**"SELECT u FROM User u WHERE "  **+** "u.username = :username AND u.password = :password"**);**  query**.**setParameter**(**"username"**,** username**);**  query**.**setParameter**(**"password"**,** password**);**  **return** **(**User**)** query**.**getSingleResult**();**  **}**  @Transactional @Override  public User updateToken**(**User user**,** String generatedToken**)** **{**  Query query **=** entityManager  **.**createQuery**(**"UPDATE User u SET u.token = :token WHERE u.id = :userId"**);**  query**.**setParameter**(**"token"**,** generatedToken**);**  query**.**setParameter**(**"userId"**,** user**.**getId**());**  query**.**executeUpdate**();**  user**.**setToken**(**generatedToken**);**  **return** user**;**  **}**  @Override  public User getUserByToken**(**String accessToken**)** **{**  Query query **=** entityManager**.**createQuery**(**"SELECT u FROM User u WHERE "  **+** "u.token = :token"**);**  query**.**setParameter**(**"token"**,** accessToken**);**  **return** **(**User**)** query**.**getSingleResult**();**  **}**  @Override  public Role getRoleById**(**String id**)** **{**  **return** **(**Role**)** entityManager**.**find**(**Role**.**class**,** id**);**  **}**  @SuppressWarnings**(**"unchecked"**)**  @Override  public EntityListWrapper**<**User**>** all**(**int max**,** int page**)** **{**  Query query **=** entityManager**.**createQuery**(**"SELECT u FROM User u"**);**  **if** **(**max **>** 0**)** **{**  query**.**setMaxResults**(**max**);**  **}**  query**.**setFirstResult**(**page **\*** max**);**  long rowcount **=** entityManager**.**createQuery**(**"SELECT count(\*) FROM User u"**,** Long**.**class**)**  **.**getSingleResult**();**  EntityListWrapper**<**User**>** users **=** **new** EntityListWrapper**<**User**>();**  users**.**setCurrentPage**(**page**);**  users**.**setEntityList**(**query**.**getResultList**());**  users**.**setLimit**(**max**);**  users**.**setRowCount**(**rowcount**);**  users**.**setTotalPage**(**PagingUtils**.**getTotalPage**(**rowcount**,** max**));**  **return** users**;**  **}**  @Override  public Role getRoleByAuth**(**String auth**)** **{**  **return** **(**Role**)** entityManager**.**createQuery**(**"SELECT u FROM Role u WHERE u.authority = ?1"**)**  **.**setParameter**(**1**,** auth**)**  **.**setMaxResults**(**1**)**  **.**getSingleResult**();**  **}**  **}** |

TaskService.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**service**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Image**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**EntityListWrapper**;**  **import** java**.**util**.**List**;**  public interface TaskService **{**  public Task get**(**String id**);**    public void remove**(**String id**);**    public List**<**Task**>** list**(**boolean isDone**);**    public String add**(**Task task**);**    public void saveImage**(**Image image**);**    public EntityListWrapper**<**Task**>** find**(**String keywords**,** int max**,** int page**);**  **}** |

TaskServiceImpl.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**service**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Image**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**EntityListWrapper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**base**.**PagingUtils**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**security**.**TokenInterceptor**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**security**.**UserService**;**  **import** java**.**util**.**List**;**  **import** javax**.**persistence**.**EntityManager**;**  **import** javax**.**persistence**.**PersistenceContext**;**  **import** javax**.**persistence**.**Query**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**security**.**core**.**context**.**SecurityContextHolder**;**  **import** org**.**springframework**.**stereotype**.**Repository**;**  **import** org**.**springframework**.**transaction**.**annotation**.**Transactional**;**  @Repository  @Transactional**(**readOnly **=** **true)**  public class TaskServiceImpl **implements** TaskService **{**    @PersistenceContext  private EntityManager entityManager**;**    @Autowired  private UserService userService**;**    @Override  @Transactional**(**readOnly**=false)**  public String add**(**Task task**)** **{**  **if** **(**""**.**equals**(**task**.**getId**()))** **{**  entityManager**.**persist**(**task**);**  **}** **else** **{**  entityManager**.**merge**(**task**);**  **}**  entityManager**.**flush**();**  **return** task**.**getId**();**  **}**  @Override  @Transactional**(**readOnly**=false)**  public void saveImage**(**Image image**)** **{**  entityManager**.**persist**(**image**);**  **}**  @SuppressWarnings**(**"unchecked"**)**  @Override  public List**<**Task**>** list**(**boolean isDone**)** **{**  **return** entityManager  **.**createQuery**(**  "SELECT t FROM Task t WHERE t.user.id = ?1 AND t.done = ?2 AND t.deleted = ?3 ORDER BY t.deadline ASC"**)**  **.**setParameter**(**1**,** TokenInterceptor**.**currentUser**().**getId**())**  **.**setParameter**(**2**,** isDone**)**  **.**setParameter**(**3**,** **false)**  **.**getResultList**();**    **}**  @Override  public Task get**(**String id**)** **{**  **return** entityManager**.**find**(**Task**.**class**,** id**);**  **}**  @Override  @Transactional**(**readOnly**=false)**  public void remove**(**String id**)** **{**  Task task **=** get**(**id**);**  task**.**setDeleted**(true);**  entityManager**.**merge**(**task**);**  **}**  @Override  public EntityListWrapper**<**Task**>** find**(**String keywords**,** int max**,** int page**)** **{**  User user **=** userService**.**findByName**(((**org**.**springframework**.**security**.**core**.**userdetails**.**User**)**SecurityContextHolder**.**getContext**().**getAuthentication**().**getPrincipal**()).**getUsername**());**  Query query **=** entityManager**.**createQuery**(**"SELECT t FROM Task t WHERE t.user.id=:user"**,** Task**.**class**);**  query**.**setParameter**(**"user"**,** user**.**getId**());**    **if** **(**max **>** 0**)** **{**  query**.**setMaxResults**(**max**);**  **}**  query**.**setFirstResult**(**page **\*** max**);**  long rowcount **=** entityManager**.**createQuery**(**"SELECT count(\*) FROM Task t WHERE t.user.id=:user"**,** Long**.**class**)**  **.**setParameter**(**"user"**,** user**.**getId**())**  **.**getSingleResult**();**  EntityListWrapper**<**Task**>** tasks **=** **new** EntityListWrapper**<**Task**>();**  tasks**.**setCurrentPage**(**page**);**  tasks**.**setEntityList**(**query**.**getResultList**());**  tasks**.**setLimit**(**max**);**  tasks**.**setRowCount**(**rowcount**);**  tasks**.**setTotalPage**(**PagingUtils**.**getTotalPage**(**rowcount**,** max**));**  **return** tasks**;**  **}**  **}** |

API.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Image**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Role**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**model**.**User**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**security**.**TokenInterceptor**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**security**.**UserService**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**service**.**TaskService**;**  **import** java**.**io**.**IOException**;**  **import** java**.**text**.**SimpleDateFormat**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**HashMap**;**  **import** java**.**util**.**Map**;**  **import** java**.**util**.**UUID**;**  **import** javax**.**persistence**.**EntityManager**;**  **import** javax**.**persistence**.**NoResultException**;**  **import** javax**.**persistence**.**PersistenceContext**;**  **import** org**.**apache**.**commons**.**logging**.**Log**;**  **import** org**.**apache**.**commons**.**logging**.**LogFactory**;**  **import** org**.**codehaus**.**jackson**.**JsonGenerationException**;**  **import** org**.**codehaus**.**jackson**.**map**.**JsonMappingException**;**  **import** org**.**codehaus**.**jackson**.**map**.**ObjectMapper**;**  **import** org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;**  **import** org**.**springframework**.**beans**.**propertyeditors**.**CustomDateEditor**;**  **import** org**.**springframework**.**http**.**HttpStatus**;**  **import** org**.**springframework**.**http**.**ResponseEntity**;**  **import** org**.**springframework**.**stereotype**.**Controller**;**  **import** org**.**springframework**.**transaction**.**annotation**.**Transactional**;**  **import** org**.**springframework**.**web**.**bind**.**WebDataBinder**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**InitBinder**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**ModelAttribute**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestMapping**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestMethod**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**RequestParam**;**  **import** org**.**springframework**.**web**.**bind**.**annotation**.**ResponseBody**;**  **import** org**.**springframework**.**web**.**multipart**.**MultipartFile**;**  @Controller  @RequestMapping**(**"/"**)**  public class API **{**  private static Log LOG **=** LogFactory**.**getLog**(**API**.**class**);**  @Autowired  private ObjectMapper mapper**;**  @Autowired  private UserService userService**;**    @Autowired  private TaskService taskService**;**    @PersistenceContext  private EntityManager entityManager**;**    @InitBinder  public void initBinder**(**WebDataBinder binder**)** **{**  SimpleDateFormat dateFormat **=** **new** SimpleDateFormat**(**"yyyy-MM-dd"**);**  dateFormat**.**setLenient**(false);**  binder**.**registerCustomEditor**(**Date**.**class**,** **new** CustomDateEditor**(**dateFormat**,** **false));**  **}**    @RequestMapping**(**value **=** "/api/task/image"**,** method **=** RequestMethod**.**POST**)**  public @ResponseBody ResponseEntity**<**String**>** upload**(**  @ModelAttribute**(**"document"**)** Image image**,**  @RequestParam**(**"file"**)** MultipartFile file**)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  LOG**.**info**(**"Name:" **+** image**.**getName**());**  LOG**.**info**(**"Desc:" **+** image**.**getDescription**());**  LOG**.**info**(**"File:" **+** file**.**getName**());**  LOG**.**info**(**"ContentType:" **+** file**.**getContentType**());**  **try** **{**  image**.**setFilename**(**file**.**getOriginalFilename**());**  image**.**setContent**(**file**.**getBytes**());**  image**.**setContentType**(**file**.**getContentType**());**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **try** **{**  taskService**.**saveImage**(**image**);**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  **}** **catch(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**    **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/api/task/list"**,** method **=** RequestMethod**.**GET**)**  public @ResponseBody ResponseEntity**<**String**>** list**(**@RequestParam**(**value **=** "done"**,** defaultValue **=** "false"**)** boolean done**)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**taskService**.**list**(**done**)),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/init"**,** method **=** RequestMethod**.**GET**)**  @Transactional  public @ResponseBody ResponseEntity**<**String**>** initApp**()** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  Role role **=** **new** Role**();**  role**.**setAuthority**(**"USER"**);**  Role role2 **=** **new** Role**();**  role2**.**setAuthority**(**"ADMIN"**);**    entityManager**.**persist**(**role2**);**  entityManager**.**persist**(**role**);**  entityManager**.**createNativeQuery**(**"alter table user\_role drop constraint user\_role\_role\_id\_key"**);**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success|"**+**role**.**getId**());**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/api/task/remove"**,** method **=** RequestMethod**.**GET**)**  public @ResponseBody ResponseEntity**<**String**>** remove**(**@RequestParam**(**value **=** "id"**,** defaultValue **=** ""**)** String id**)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  **try** **{**  taskService**.**remove**(**id**);**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  **}catch(**Throwable e**){**  map**.**put**(**"statusCode"**,** "02"**);**  map**.**put**(**"statusMessage"**,** "Persist Error"**);**  LOG**.**error**(**"E"**,** e**);**  **}**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/api/task/add"**,** method **=** RequestMethod**.**POST**)**  public @ResponseBody ResponseEntity**<**String**>** addTask**(**@ModelAttribute Task task**)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  **try** **{**  task**.**setUser**(**TokenInterceptor**.**currentUser**());**  LOG**.**info**(**"Persisting Task ["**+**task**.**getId**()+**"] "**+**task**.**toString**());**  String taskId **=** **null;**  **if** **(**task**.**getId**()** **!=** **null)** **{**  taskService**.**add**(**task**);**  Task t **=** taskService**.**get**(**task**.**getId**());**  **if** **(!**""**.**equals**(**task**.**getContent**()))**  t**.**setContent**(**task**.**getContent**());**  **if** **(**task**.**getDeadline**()** **!=** **null)**  t**.**setDeadline**(**task**.**getDeadline**());**  taskId **=** taskService**.**add**(**t**);**  **}** **else** **{**  taskId **=** taskService**.**add**(**task**);**  **}**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  map**.**put**(**"id"**,** taskId**);**  **}** **catch** **(**Throwable e**)** **{**  map**.**put**(**"statusCode"**,** "02"**);**  map**.**put**(**"statusMessage"**,** "Persist Error"**);**  map**.**put**(**"id"**,** ""**);**  LOG**.**error**(**"E"**,** e**);**  **}**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/register"**,** method **=** RequestMethod**.**POST**)**  public @ResponseBody ResponseEntity**<**String**>** registerPost**(**@ModelAttribute User user**)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  **try** **{**  LOG**.**info**(**"Register User "**+**user**.**getUsername**());**  Role role **=** userService**.**getRoleByAuth**(**"USER"**);**  LOG**.**info**(**"Seeting role "**+**role**.**getId**()+**":"**+**role**.**getAuthority**());**  // user.getUserSecurityRoleEntity().add(role);  String generatedToken **=** UUID**.**randomUUID**().**toString**().**replace**(**"-"**,** ""**);**  user**.**setToken**(**generatedToken**);**  userService**.**addUser**(**user**);**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  map**.**put**(**"secretToken"**,** user**.**getToken**());**  **}** **catch** **(**Throwable e**)** **{**  **if(**e **instanceof** org**.**springframework**.**dao**.**DataIntegrityViolationException**){**  map**.**put**(**"statusCode"**,** "03"**);**  map**.**put**(**"statusMessage"**,** "Username or Email Already Exist"**);**  **}else{**  map**.**put**(**"statusCode"**,** "02"**);**  map**.**put**(**"statusMessage"**,** "Persist Error"**);**  **}**  LOG**.**error**(**"E"**,** e**);**  **}**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**    @RequestMapping**(**value **=** "/login/token"**,** method **=** RequestMethod**.**POST**)**  public @ResponseBody ResponseEntity**<**String**>** generateToken**(**  @RequestParam**(**"username"**)** String username**,**  @RequestParam**(**"password"**)** String password  **)** **throws** JsonGenerationException**,** JsonMappingException**,** IOException **{**  Map**<**String**,** String**>** map **=** **new** HashMap**<**String**,** String**>();**  User user **=** **null;**  **try** **{**  user **=** userService**.**getUser**(**username**,** password**);**  **if** **(**user**.**getToken**()** **==** **null)** **{**  String generatedToken **=** UUID**.**randomUUID**().**toString**().**replace**(**"-"**,** ""**);**  LOG**.**info**(**"Generated security token: " **+** generatedToken**);**  user **=** userService**.**updateToken**(**user**,** generatedToken**);**  **}**  LOG**.**info**(**"Generated security token already exist: "**+** user**.**getToken**());**  map**.**put**(**"statusCode"**,** "00"**);**  map**.**put**(**"statusMessage"**,** "Success"**);**  map**.**put**(**"secretToken"**,** user**.**getToken**());**  **}** **catch** **(**NoResultException e**)** **{**  map**.**put**(**"statusCode"**,** "01"**);**  map**.**put**(**"statusMessage"**,** "Bad credentials"**);**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**UNAUTHORIZED**);**  **}**  **return** **new** ResponseEntity**<**String**>(**mapper**.**writeValueAsString**(**map**),** HttpStatus**.**OK**);**  **}**  **}** |

applicationContext.xml

|  |
| --- |
| <?xml version=**"1.0"** encoding=**"UTF-8"**?>  <beans xmlns=**"http://www.springframework.org/schema/beans"**  xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"** xmlns:p=**"http://www.springframework.org/schema/p"**  xmlns:aop=**"http://www.springframework.org/schema/aop"** xmlns:context=**"http://www.springframework.org/schema/context"**  xmlns:jee=**"http://www.springframework.org/schema/jee"** xmlns:tx=**"http://www.springframework.org/schema/tx"**  xmlns:mvc=**"http://www.springframework.org/schema/mvc"**  xsi:schemaLocation=**"**  **http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-2.5.xsd**  **http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.5.xsd**  **http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-2.5.xsd**  **http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-2.5.xsd**  **http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-2.5.xsd**  **http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd"**>  <context:component-scan base-package=**"id.ac.gunadarma"** />  <context:property-placeholder location=**"classpath\*:jdbc.properties"** />    <context:annotation-config />  <bean id=**"dataSource"** class=**"com.mchange.v2.c3p0.ComboPooledDataSource"** destroy-method=**"close"**>  <property name=**"driverClass"** value=**"${jdbc.driverClass}"** />  <property name=**"jdbcUrl"** value=**"${jdbc.url}"** />  <property name=**"user"** value=**"${jdbc.username}"** />  <property name=**"password"** value=**"${jdbc.password}"** />  </bean>  <bean id=**"entityManagerFactory"**  class=**"org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean"**  p:dataSource-ref=**"dataSource"**>  <property name=**"jpaVendorAdapter"**>  <bean class=**"org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"**>  <property name=**"database"** value=**"POSTGRESQL"** />  <property name=**"showSql"** value=**"${jdbc.showSql}"** />  <property name=**"generateDdl"** value=**"true"**/>  </bean>  </property>  <property name=**"persistenceXmlLocation"** value=**"classpath:/persistence.xml"** />  </bean>  <bean id=**"multipartResolver"**  class=**"org.springframework.web.multipart.commons.CommonsMultipartResolver"**>  <property name=**"maxUploadSize"** value=**"10000000"** />  </bean>    <bean id=**"transactionManager"** class=**"org.springframework.orm.jpa.JpaTransactionManager"**  p:entityManagerFactory-ref=**"entityManagerFactory"** />  <bean id=**"objectMapper"** class=**"org.codehaus.jackson.map.ObjectMapper"**/>    </beans> |

applicationContextSecurity.xml

|  |
| --- |
| <beans:beans xmlns=**"http://www.springframework.org/schema/security"**  xmlns:beans=**"http://www.springframework.org/schema/beans"** xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**  xsi:schemaLocation=**"http://www.springframework.org/schema/beans**  **http://www.springframework.org/schema/beans/spring-beans-3.0.xsd**  **http://www.springframework.org/schema/security**  **http://www.springframework.org/schema/security/spring-security-3.0.3.xsd"**>  <http auto-config=**"true"**>  <intercept-url pattern=**"/user\*"** access=**"ROLE\_GOD"** />  <form-login login-page=**"/login"** default-target-url=**"/index"** authentication-failure-url=**"/loginfailed"** />  <logout logout-success-url=**"/logout"** />  </http>  <authentication-manager>  <authentication-provider user-service-ref=**"userDetailsService"**>  <password-encoder hash=**"plaintext"** />  </authentication-provider>  </authentication-manager>  </beans:beans> |

jdbc.properties

|  |
| --- |
| jdbc.driverClass**=**org.postgresql.Driver  jdbc.url**=**jdbc:postgresql://ec2-184-72-238-68.compute-1.amazonaws.com:5432/d39if5vqr6bqsc?ssl=true&sslfactory=org.postgresql.ssl.NonValidatingFactory  jdbc.username**=**vvuwptwvsbxkfb  jdbc.password**=**XEVJ2nKd20WVJrhH9hHZeBRYlC  jdbc.showSql**=**false |

log4j.properties

|  |
| --- |
| log4j.rootCategory**=**INFO, stdout, logfile  log4j.appender.stdout**=**org.apache.log4j.ConsoleAppender  log4j.appender.stdout.layout**=**org.apache.log4j.PatternLayout  log4j.appender.stdout.layout.ConversionPattern**=**%d %p [%c] - <%m>%n  log4j.appender.logfile**=**org.apache.log4j.RollingFileAppender  log4j.appender.logfile.File**=**/tmp/clinicComponet.log  log4j.appender.logfile.MaxFileSize**=**1MB  # Keep three backup files  log4j.appender.logfile.MaxBackupIndex**=**3  log4j.appender.logfile.layout**=**org.apache.log4j.PatternLayout  #Pattern to output : date priority [category] - <message>line\_separator  log4j.appender.logfile.layout.ConversionPattern**=**%d %p [%c] - <%m>%n  log4j.logger.org.springframework.web.servlet.handler.AbstractUrlHandlerMapping**=**ERROR  log4j.logger.org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping**=**ERROR  log4j.logger.org.springframework.web.servlet.PageNotFound**=**ERROR |

messages.properties

|  |
| --- |
| AbstractUserDetailsAuthenticationProvider.badCredentials**=**Invalid username or password |

mvc-dispatcher-servlet.xml

|  |
| --- |
| <beans xmlns=**"http://www.springframework.org/schema/beans"**  xmlns:context=**"http://www.springframework.org/schema/context"**  xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"** xmlns:mvc=**"http://www.springframework.org/schema/mvc"**  xmlns:tx=**"http://www.springframework.org/schema/tx"**  xsi:schemaLocation=**"**  **http://www.springframework.org/schema/beans**  **http://www.springframework.org/schema/beans/spring-beans-3.0.xsd**  **http://www.springframework.org/schema/context**  **http://www.springframework.org/schema/context/spring-context-3.0.xsd**  **http://www.springframework.org/schema/mvc**  **http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd**  **http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-2.5.xsd"**>  <context:component-scan base-package=**"id.ac.gunadarma"** />    <mvc:annotation-driven />    <tx:annotation-driven transaction-manager=**"transactionManager"** />  <mvc:resources mapping=**"/resources/\*\*"** location=**"/public-resources/"** />  <bean id=**"fmXmlEscape"** class=**"freemarker.template.utility.XmlEscape"** />  <bean id=**"freemarkerConfig"**  class=**"org.springframework.web.servlet.view.freemarker.FreeMarkerConfigurer"**>  <property name=**"templateLoaderPath"**>  <value>**/WEB-INF/pages/**</value>  </property>  <property name=**"freemarkerVariables"**>  <map>  <entry key=**"xml\_escape"** value-ref=**"fmXmlEscape"** />  </map>  </property>  <property name=**"freemarkerSettings"**>  <props>  <prop key=**"template\_update\_delay"**>**3**</prop>  </props>  </property>  </bean>  <bean  class=**"org.springframework.web.servlet.view.ContentNegotiatingViewResolver"**>  <property name=**"mediaTypes"**>  <map>  <entry key=**"html"** value=**"text/html"** />  <entry key=**"ftl"** value=**"text/html"** />  <entry key=**"json"** value=**"application/json"** />  </map>  </property>  <property name=**"favorPathExtension"** value=**"true"** />  <property name=**"defaultViews"**>  <list>  <bean  class=**"org.springframework.web.servlet.view.json.MappingJacksonJsonView"** />  </list>  </property>  <property name=**"viewResolvers"**>  <list>  <bean  class=**"org.springframework.web.servlet.view.freemarker.FreeMarkerViewResolver"**>  <property name=**"cache"** value=**"true"** />  <property name=**"order"** value=**"1"** />  <property name=**"prefix"** value=**"/"** />  <property name=**"suffix"** value=**".ftl"** />  <property name=**"contentType"** value=**"text/html;charset=UTF-8"** />  <property name=**"exposeSpringMacroHelpers"** value=**"true"** />  <property name=**"requestContextAttribute"** value=**"rc"** />  <property name=**"exposeSessionAttributes"** value=**"true"** />  </bean>  </list>  </property>  </bean>  <bean id=**"messageSource"**  class=**"org.springframework.context.support.ResourceBundleMessageSource"**>  <property name=**"basenames"**>  <list>  <value>**messages**</value>  </list>  </property>  </bean>  <mvc:interceptors>  <mvc:interceptor>  <mvc:mapping path=**"/api/\*\*"** />  <bean class=**"id.ac.gunadarma.tugasku.security.TokenInterceptor"** />  </mvc:interceptor>  </mvc:interceptors>  </beans> |

persistence.xml

|  |
| --- |
| <persistence xmlns=**"http://java.sun.com/xml/ns/persistence"**  xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**  xsi:schemaLocation=**"http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence\_1\_0.xsd"**  version=**"1.0"**>  <persistence-unit name=**"default"** transaction-type=**"RESOURCE\_LOCAL"**>  <properties>  <property name=**"hibernate.dialect"** value=**"org.hibernate.dialect.PostgreSQLDialect"** />  <property name=**"hibernate.hbm2ddl.auto"** value=**"update"** />  <property name=**"hibernate.show.sql"** value=**"true"** />  </properties>  <class>**id.ac.gunadarma.tugasku.model.User**</class>  <class>**id.ac.gunadarma.tugasku.model.Role**</class>  <class>**id.ac.gunadarma.tugasku.model.Task**</class>  <class>**id.ac.gunadarma.tugasku.model.Image**</class>  </persistence-unit>  </persistence> |

index.ftl

|  |
| --- |
| <!DOCTYPE html>  <html lang=**"en"**>  <head>  <meta charset=**"utf-8"**>  <title>**Home** &middot; **Management Tugas**</title>  <meta name=**"viewport"** content=**"width=device-width, initial-scale=1.0"**>  <meta name=**"description"** content=**""**>  <meta name=**"author"** content=**""**>  <!-- Le styles -->  <link href=**"${rc.contextPath}/resources/css/bootstrap.css"**  rel=**"stylesheet"**>  <style type=**"text/css"**>  **body {**  **padding-top: 20px;**  **padding-bottom: 40px;**  **}**  **/\* Custom container \*/**  **.container-narrow {**  **margin: 0 auto;**  **max-width: 700px;**  **}**  **.container-narrow>hr {**  **margin: 30px 0;**  **}**  **/\* Main marketing message and sign up button \*/**  **.jumbotron {**  **margin: 60px 0;**  **text-align: center;**  **}**  **.jumbotron h1 {**  **font-size: 72px;**  **line-height: 1;**  **}**  **.jumbotron .btn {**  **font-size: 21px;**  **padding: 14px 24px;**  **}**  **/\* Supporting marketing content \*/**  **.marketing {**  **margin: 60px 0;**  **}**  **.marketing p+h4 {**  **margin-top: 28px;**  **}**  </style>  <link href=**"${rc.contextPath}/resources/css/bootstrap-responsive.css"**  rel=**"stylesheet"**>  <!-- HTML5 shim, for IE6-8 support of HTML5 elements -->  <!--[if lt IE 9]>  <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>  <![endif]-->  </head>  <body>  <div class=**"container-narrow"**>  <h3 class=**"muted"**>**Management Tugas**</h3>  <div class=**"masthead"**>  <ul class=**"nav nav-pills pull-right"**>  <li class=**"active"**><a href=**"${rc.contextPath}/game"**>**Tasks**</a></li>    </ul>  <div class=**"btn-group"**>  <a class=**"btn btn-primary"** href=**"#"**><i class=**"icon-user icon-white"**></i> **User**</a>  <a class=**"btn btn-primary dropdown-toggle"** data-toggle=**"dropdown"** href=**"#"**><span class=**"caret"**></span></a>  <ul class=**"dropdown-menu"**>  <li><a href=**"${rc.contextPath}/j\_spring\_security\_logout"**><i class=**"icon-share"**></i> **Logout**</a></li>  </ul>  </div>  </div>    <hr>    <table class=**"table table-bordered"**>  <thead>  <tr>  <th>**#**</th>  <th>**Content**</th>  <th>**Deadline**</th>  <th>**Task Done?**</th>  </tr>  </thead>  <tbody>  <#assign no = (max \* page) - max + 1>  <#list tasks.entityList as c>  <tr>  <td>**${no}**</td>  <td>**${c.content}**</td>  <td>**${c.deadline}**</td>  <td>**${c.done?string("yes", "no")}**</td>  </tr>  <#assign no = no + 1>  </#list>  </tbody>  </table>  <span class=**"label pull-left"**>**Found ${tasks.rowCount} row(s)**</span>  <div class=**"btn-group pull-right"**>  <#if page &gt; 1>  <a class=**"btn"** href=**"${rc.contextPath}?max=${max}&page=1"**><i class=**"icon-fast-backward"**></i>&nbsp;</a>  <a class=**"btn"** href=**"${rc.contextPath}?max=${max}&page=${page - 1}"**><i class=**"icon-backward"**></i>&nbsp;</a>  </#if>  <a class=**"btn disabled"**>**${page} of ${tasks.totalPage}**</a>  <#if page &lt; tasks.totalPage>  <a class=**"btn"** href=**"${rc.contextPath}?max=${max}&page=${page + 1}"**>&nbsp;<i class=**"icon-forward"**></i></a>  <a class=**"btn"** href=**"${rc.contextPath}?max=${max}&page=${tasks.totalPage}"**>&nbsp;<i class=**"icon-fast-forward"**></i></a>  </#if>  </div>  </div>  <script src=**"${rc.contextPath}/resources/js/jquery.js"**></script>  <script src=**"${rc.contextPath}/resources/js/bootstrap.min.js"**></script>  </body>  </html> |

login.ftl

|  |
| --- |
| <!DOCTYPE html>  <html lang=**"en"**>  <head>  <meta charset=**"utf-8"**>  <title>**Sign in** &middot; **Twitter Bootstrap**</title>  <meta name=**"viewport"** content=**"width=device-width, initial-scale=1.0"**>  <meta name=**"description"** content=**""**>  <meta name=**"author"** content=**""**>  <!-- Le styles -->  <link href=**"${rc.contextPath}/resources/css/bootstrap.css"** rel=**"stylesheet"**>  <style type=**"text/css"**>  **body {**  **padding-top: 40px;**  **padding-bottom: 40px;**  **background-color: #f5f5f5;**  **}**  **.form-signin {**  **max-width: 300px;**  **padding: 19px 29px 29px;**  **margin: 0 auto 20px;**  **background-color: #fff;**  **border: 1px solid #e5e5e5;**  **-webkit-border-radius: 5px;**  **-moz-border-radius: 5px;**  **border-radius: 5px;**  **-webkit-box-shadow: 0 1px 2px rgba(0,0,0,.05);**  **-moz-box-shadow: 0 1px 2px rgba(0,0,0,.05);**  **box-shadow: 0 1px 2px rgba(0,0,0,.05);**  **}**  **.form-signin .form-signin-heading,**  **.form-signin .checkbox {**  **margin-bottom: 10px;**  **}**  **.form-signin input[type="text"],**  **.form-signin input[type="password"] {**  **font-size: 16px;**  **height: auto;**  **margin-bottom: 15px;**  **padding: 7px 9px;**  **}**  </style>  <link href=**"${rc.contextPath}/resources/css/bootstrap-responsive.css"** rel=**"stylesheet"**>  <!-- HTML5 shim, for IE6-8 support of HTML5 elements -->  <!--[if lt IE 9]>  <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>  <![endif]-->  </head>  <body>  <div class=**"container"**>  <form name=**'f'** action=**"${rc.contextPath}/j\_spring\_security\_check"** method=**'POST'** class=**"form-signin"**>  <#if not?? || empty?? || error?? >  <div class=**"alert alert-error"**>  **Login Error : ${Session["SPRING\_SECURITY\_LAST\_EXCEPTION"].message}**  </div>  </#if>  <#if success=="true" >  <div class=**"alert alert-success"**>  **Registered!! Please Sign In Below**  </div>  </#if>  <h2 class=**"form-signin-heading"**>**Please sign in**</h2>  <input type=**'text'** name=**'j\_username'** value=**''** class=**"input-block-level"** placeholder=**"Username"**>  <input type=**'password'** name=**'j\_password'** class=**"input-block-level"** placeholder=**"Password"**>  <label class=**"checkbox"**>  <input type=**"checkbox"** value=**"remember-me"**> **Remember me**  </label>  <button class=**"btn btn-primary pull-right"** type=**"submit"**>**Sign in**</button>  <br>  <br>  <p class=**"pull-right"**>**or** <a href=**"${rc.contextPath}/registerweb"**>**Register**</a>  <br>  </form>  </div>  <script src=**"${rc.contextPath}/resources/js/jquery.js"**></script>  <script src=**"${rc.contextPath}/resources/js/bootstrap.min.js"**></script>    </body>  </html> |

register.ftl

|  |
| --- |
| <!DOCTYPE html>  <html lang=**"en"**>  <head>  <meta charset=**"utf-8"**>  <title>**User** &middot; **MyGameList**</title>  <meta name=**"viewport"** content=**"width=device-width, initial-scale=1.0"**>  <meta name=**"description"** content=**""**>  <meta name=**"author"** content=**""**>  <!-- Le styles -->  <link href=**"${rc.contextPath}/resources/css/bootstrap.css"**  rel=**"stylesheet"**>  <style type=**"text/css"**>  **body {**  **padding-top: 20px;**  **padding-bottom: 40px;**  **}**  **/\* Custom container \*/**  **.container-narrow {**  **margin: 0 auto;**  **max-width: 700px;**  **}**  **.container-narrow>hr {**  **margin: 30px 0;**  **}**  **/\* Main marketing message and sign up button \*/**  **.jumbotron {**  **margin: 60px 0;**  **text-align: center;**  **}**  **.jumbotron h1 {**  **font-size: 72px;**  **line-height: 1;**  **}**  **.jumbotron .btn {**  **font-size: 21px;**  **padding: 14px 24px;**  **}**  **/\* Supporting marketing content \*/**  **.marketing {**  **margin: 60px 0;**  **}**  **.marketing p+h4 {**  **margin-top: 28px;**  **}**  </style>  <link href=**"${rc.contextPath}/resources/css/bootstrap-responsive.css"**  rel=**"stylesheet"**>  <!-- HTML5 shim, for IE6-8 support of HTML5 elements -->  <!--[if lt IE 9]>  <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>  <![endif]-->  </head>  <body>  <div class=**"container-narrow"**>  <div class=**"masthead"**>  <h3 class=**"muted"**>**Manajemen Tugas**</h3>  </div>    <hr>    <form class=**"form-horizontal"** action=**""** method=**"post"**>  <fieldset>  <div class=**"control-group"**>  <label class=**"control-label"** for=**"input01"**>**First Name**</label>  <div class=**"controls"**>  <input name=**"fname"** type=**"text"** placeholder=**""** class=**"input-xlarge"**>  <p class=**"help-block"**></p>  </div>  </div>  <div class=**"control-group"**>  <label class=**"control-label"** for=**"input01"**>**Last Name**</label>  <div class=**"controls"**>  <input name=**"lname"** type=**"text"** placeholder=**""** class=**"input-xlarge"**>  <p class=**"help-block"**></p>  </div>  </div>  <div class=**"control-group"**>  <label class=**"control-label"** for=**"input01"**>**Username**</label>  <div class=**"controls"**>  <input name=**"username"** type=**"text"** placeholder=**""** class=**"input-xlarge"**>  </div>  </div>  <div class=**"control-group"**>  <label class=**"control-label"** for=**"input01"**>**Password**</label>  <div class=**"controls"**>  <input name=**"password"** type=**"password"** placeholder=**""** class=**"input-xlarge"**>  </div>  </div>  <div class=**"control-group"**>  <label class=**"control-label"** for=**"input01"**>**Email**</label>  <div class=**"controls"**>  <input name=**"email"** type=**"text"** placeholder=**""** class=**"input-xlarge"**>  </div>  </div>  <button class=**"btn btn-primary pull-right"** type=**"submit"**>&nbsp;&nbsp;&nbsp;**Register**&nbsp;&nbsp;&nbsp;</button>  <br>  </fieldset>  </form>  </div>  <script src=**"${rc.contextPath}/resources/js/jquery.js"**></script>  <script src=**"${rc.contextPath}/resources/js/bootstrap.min.js"**></script>  </body>  </html> |

web.xml

|  |
| --- |
| <web-app xmlns=**"http://java.sun.com/xml/ns/javaee"** xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**  xsi:schemaLocation=**"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"**  version=**"2.5"**>  <display-name>**TugasKu Web Application**</display-name>  <!-- Spring MVC -->  <servlet>  <servlet-name>**mvc-dispatcher**</servlet-name>  <servlet-class>**org.springframework.web.servlet.DispatcherServlet**</servlet-class>  <init-param>  <param-name>**contextConfigLocation**</param-name>  <param-value>**classpath:mvc-dispatcher-servlet.xml**</param-value>  </init-param>  <load-on-startup>**1**</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>**mvc-dispatcher**</servlet-name>  <url-pattern>**/**</url-pattern>  </servlet-mapping>  <listener>  <listener-class>**org.springframework.web.context.ContextLoaderListener**</listener-class>  </listener>  <context-param>  <param-name>**contextConfigLocation**</param-name>  <param-value>  **classpath:/applicationContext\*.xml**  </param-value>  </context-param>  <filter>  <filter-name>  **OpenEntityManagerInViewFilter**  </filter-name>  <filter-class>  **org.springframework.orm.jpa.support.OpenEntityManagerInViewFilter**  </filter-class>  <init-param>  <param-name>**entityManagerFactoryBeanName**</param-name>  <param-value>**entityManagerFactory**</param-value>  </init-param>  <init-param>  <param-name>**flushMode**</param-name>  <param-value>**AUTO**</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>**OpenEntityManagerInViewFilter**</filter-name>  <url-pattern>**/\***</url-pattern>  </filter-mapping>  <!-- Spring Security -->  <filter>  <filter-name>**springSecurityFilterChain**</filter-name>  <filter-class>**org.springframework.web.filter.DelegatingFilterProxy**</filter-class>  </filter>  <filter-mapping>  <filter-name>**springSecurityFilterChain**</filter-name>  <url-pattern>**/\***</url-pattern>  </filter-mapping>  <container-descriptor>  <show-archived-real-path-enabled>**true**</show-archived-real-path-enabled>  </container-descriptor>  </web-app> |

AndroidManifest.xml

|  |
| --- |
| <?xml version=**"1.0"** encoding=**"utf-8"**?>  <!--  Copyright 2012 The Android Open Source Project  Licensed under the Apache License, Version 2.0 (the "License");  you may not use this file except in compliance with the License.  You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing, software  distributed under the License is distributed on an "AS IS" BASIS,  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  See the License for the specific language governing permissions and  limitations under the License.  -->  <manifest xmlns:android=**"http://schemas.android.com/apk/res/android"**  package=**"id.ac.gunadarma.tugasku"**  android:versionCode=**"1"**  android:versionName=**"1.0"** >  <uses-sdk  android:minSdkVersion=**"14"**  android:targetSdkVersion=**"18"** />  <uses-permission android:name=**"android.permission.USE\_CREDENTIALS"**/>  <uses-permission android:name=**"android.permission.GET\_ACCOUNTS"**/>  <uses-permission android:name=**"android.permission.MANAGE\_ACCOUNTS"**/>  <!-- Authenticator -->  <uses-permission android:name=**"android.permission.INTERNET"**/>  <uses-permission android:name=**"android.permission.AUTHENTICATE\_ACCOUNTS"**/>  <!-- Sync Adapter -->  <uses-permission android:name=**"android.permission.READ\_SYNC\_STATS"** />  <uses-permission android:name=**"android.permission.READ\_SYNC\_SETTINGS"** />  <uses-permission android:name=**"android.permission.WRITE\_SYNC\_SETTINGS"** />    <uses-permission android:name=**"android.permission.WAKE\_LOCK"**></uses-permission>  <uses-permission android:name=**"android.permission.RECEIVE\_BOOT\_COMPLETED"**></uses-permission>    <application  android:allowBackup=**"true"**  android:icon=**"@drawable/ic\_launcher"**  android:label=**"@string/app\_name"**  android:theme=**"@style/Theme.Sherlock.Light"** >    <activity  android:name=**"id.ac.gunadarma.tugasku.FragmentTabsPager"**  android:label=**"@string/app\_name"**  >  <intent-filter>  <action android:name=**"android.intent.action.MAIN"** />  <category android:name=**"android.intent.category.LAUNCHER"** />  </intent-filter>  </activity>    <activity android:name=**"id.ac.gunadarma.tugasku.TaskFormFragmentActivity"** />  <activity android:name=**"id.ac.gunadarma.tugasku.PreferenceActivity"** />  <activity android:name=**"id.ac.gunadarma.tugasku.LoginActivity"** />    <provider android:authorities=**"id.ac.gunadarma.tugasku.provider"**  android:name=**".db.TaskContentProvider"**  android:label=**"@string/provider\_name"**  android:exported=**"false"**/>  <service  android:name=**".sync.TaskSyncService"** android:exported=**"true"** >  <intent-filter>  <action android:name=**"android.content.SyncAdapter"** />  </intent-filter>  <meta-data android:name=**"android.content.SyncAdapter"**  android:resource=**"@xml/sync\_adapter"** />  </service>  <service android:name=**".account.TaskAuthenticatorService"**>  <intent-filter>  <action android:name=**"android.accounts.AccountAuthenticator"** />  </intent-filter>  <meta-data android:name=**"android.accounts.AccountAuthenticator"**  android:resource=**"@xml/authenticator"** />  </service>    <receiver android:name=**".helper.alarm.MemoAlarmReceiver"**></receiver>    <receiver android:name=**".helper.alarm.MemoBootReceiver"**  android:enabled=**"false"**>  <intent-filter>  <action android:name=**"android.intent.action.BOOT\_COMPLETED"**></action>  </intent-filter>  </receiver>  </application>  </manifest> |

AccountGeneral.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**account**;**  public class AccountGeneral **{**    public static final ServerAuthenticate sServerAuthenticate **=** **new** TaskServerAuthenticate**();**    public static final String ACCOUNT\_TYPE **=** "id.ac.gunadarma.tugasku"**;**  public static final String AUTHTOKEN\_TYPE\_FULL\_ACCESS **=** "Full access"**;**  public static final String USERDATA\_USER\_OBJ\_ID **=** "USERDATA\_USER\_OBJ\_ID"**;**  **}** |

ServerAuthenticate.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**account**;**  public interface ServerAuthenticate **{**    public String userSignUp**(**final String name**,** final String username**,** final String email**,** final String pass**,** String authType**)** **throws** Exception**;**    public String userSignIn**(**final String user**,** final String pass**,** String authType**)** **throws** Exception**;**    **}** |

TaskAuthenticator.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**account**;**  **import** static android**.**accounts**.**AccountManager**.**KEY\_BOOLEAN\_RESULT**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**LoginActivity**;**  **import** android**.**accounts**.**AbstractAccountAuthenticator**;**  **import** android**.**accounts**.**Account**;**  **import** android**.**accounts**.**AccountAuthenticatorResponse**;**  **import** android**.**accounts**.**AccountManager**;**  **import** android**.**accounts**.**NetworkErrorException**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**text**.**TextUtils**;**  **import** android**.**util**.**Log**;**  public class TaskAuthenticator **extends** AbstractAccountAuthenticator **{**  private String TAG **=** "TaskAuthenticator"**;**  private final Context mContext**;**  public TaskAuthenticator**(**Context context**)** **{**  **super(**context**);**  **this.**mContext **=** context**;**  **}**  @Override  public Bundle addAccount**(**AccountAuthenticatorResponse response**,** String accountType**,** String authTokenType**,** String**[]** requiredFeatures**,** Bundle options**)** **throws** NetworkErrorException **{**  final Intent intent **=** **new** Intent**(**mContext**,** LoginActivity**.**class**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_ACCOUNT\_TYPE**,** accountType**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_AUTH\_TYPE**,** authTokenType**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_IS\_ADDING\_NEW\_ACCOUNT**,** **true);**  intent**.**putExtra**(**AccountManager**.**KEY\_ACCOUNT\_AUTHENTICATOR\_RESPONSE**,** response**);**  final Bundle bundle **=** **new** Bundle**();**  bundle**.**putParcelable**(**AccountManager**.**KEY\_INTENT**,** intent**);**  **return** bundle**;**  **}**  @Override  public Bundle getAuthToken**(**AccountAuthenticatorResponse response**,** Account account**,** String authTokenType**,** Bundle options**)** **throws** NetworkErrorException **{**  final AccountManager am **=** AccountManager**.**get**(**mContext**);**  String authToken **=** am**.**peekAuthToken**(**account**,** authTokenType**);**  Log**.**d**(**"Task"**,** TAG **+** "> peekAuthToken returned - " **+** authToken**);**  **if** **(**TextUtils**.**isEmpty**(**authToken**))** **{**  final String password **=** am**.**getPassword**(**account**);**  **if** **(**password **!=** **null)** **{**  **try** **{**  Log**.**d**(**"udinic"**,** TAG **+** "> re-authenticating with the existing password"**);**  authToken **=** AccountGeneral**.**sServerAuthenticate**.**userSignIn**(**account**.**name**,** password**,** authTokenType**);**  **}** **catch** **(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**  **}**  **}**  **if** **(!**TextUtils**.**isEmpty**(**authToken**))** **{**  final Bundle result **=** **new** Bundle**();**  result**.**putString**(**AccountManager**.**KEY\_ACCOUNT\_NAME**,** account**.**name**);**  result**.**putString**(**AccountManager**.**KEY\_ACCOUNT\_TYPE**,** account**.**type**);**  result**.**putString**(**"MONYET"**,** "OK"**);**  result**.**putString**(**AccountManager**.**KEY\_AUTHTOKEN**,** authToken**);**  **return** result**;**  **}**  final Intent intent **=** **new** Intent**(**mContext**,** LoginActivity**.**class**);**  intent**.**putExtra**(**AccountManager**.**KEY\_ACCOUNT\_AUTHENTICATOR\_RESPONSE**,** response**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_ACCOUNT\_TYPE**,** account**.**type**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_AUTH\_TYPE**,** authTokenType**);**  intent**.**putExtra**(**LoginActivity**.**ARG\_ACCOUNT\_NAME**,** account**.**name**);**  final Bundle bundle **=** **new** Bundle**();**  bundle**.**putParcelable**(**AccountManager**.**KEY\_INTENT**,** intent**);**  **return** bundle**;**  **}**  @Override  public String getAuthTokenLabel**(**String authTokenType**)** **{**  **return** authTokenType **+** "Access Task"**;**  **}**  @Override  public Bundle hasFeatures**(**AccountAuthenticatorResponse responseu**,** Account account**,** String**[]** features**)** **throws** NetworkErrorException **{**  final Bundle result **=** **new** Bundle**();**  result**.**putBoolean**(**KEY\_BOOLEAN\_RESULT**,** **false);**  **return** result**;**  **}**  @Override  public Bundle editProperties**(**AccountAuthenticatorResponse response**,** String accountType**)** **{**  **return** **null;**  **}**  @Override  public Bundle confirmCredentials**(**AccountAuthenticatorResponse response**,** Account account**,** Bundle options**)** **throws** NetworkErrorException **{**  **return** **null;**  **}**  @Override  public Bundle updateCredentials**(**AccountAuthenticatorResponse response**,** Account account**,** String authTokenType**,** Bundle options**)** **throws** NetworkErrorException **{**  **return** **null;**  **}**  **}** |

TaskAuthenticatorService.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**account**;**  **import** android**.**app**.**Service**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**IBinder**;**  public class TaskAuthenticatorService **extends** Service **{**  @Override  public IBinder onBind**(**Intent intent**)** **{**  **return** **new** TaskAuthenticator**(this).**getIBinder**();**  **}**  **}** |

TaskServerAuthenticate.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**account**;**  **import** java**.**io**.**IOException**;**  **import** java**.**io**.**Serializable**;**  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**List**;**  **import** org**.**apache**.**http**.**HttpResponse**;**  **import** org**.**apache**.**http**.**NameValuePair**;**  **import** org**.**apache**.**http**.**client**.**entity**.**UrlEncodedFormEntity**;**  **import** org**.**apache**.**http**.**client**.**methods**.**HttpPost**;**  **import** org**.**apache**.**http**.**impl**.**client**.**DefaultHttpClient**;**  **import** org**.**apache**.**http**.**message**.**BasicNameValuePair**;**  **import** org**.**apache**.**http**.**util**.**EntityUtils**;**  **import** android**.**util**.**Log**;**  **import** com**.**google**.**gson**.**Gson**;**  public class TaskServerAuthenticate **implements** ServerAuthenticate **{**    @Override  public String userSignUp**(**String name**,** String username**,** String email**,** String pass**,** String authType**)** **throws** Exception **{**  String url **=** "http://tugasku.herokuapp.com/register"**;**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  HttpPost httpPost **=** **new** HttpPost**(**url**);**  List**<**NameValuePair**>** nameValuePairs **=** **new** ArrayList**<**NameValuePair**>();**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"first\_name"**,** name**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"username"**,** username**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"password"**,** username**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"email"**,** email**));**  httpPost**.**setEntity**(new** UrlEncodedFormEntity**(**nameValuePairs**));**  String authtoken **=** **null;**  **try** **{**  HttpResponse response **=** httpClient**.**execute**(**httpPost**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  CredentialsResponse registerResponse **=** **new** Gson**().**fromJson**(**responseString**,** CredentialsResponse**.**class**);**  **if** **(**registerResponse**.**statusCode **!=** 0**)** **{**  **throw** **new** Exception**(**"Error signing-in ["**+**registerResponse**.**statusCode**+**"] - " **+** registerResponse**.**statusMessage**);**  **}** **else** **{**  authtoken **=** registerResponse**.**secretToken**;**  **}**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **return** authtoken**;**  **}**  @Override  public String userSignIn**(**String username**,** String password**,** String authType**)** **throws** Exception **{**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  String url **=** "http://tugasku.herokuapp.com/login/token"**;**  HttpPost httpPost **=** **new** HttpPost**(**url**);**    List**<**NameValuePair**>** nameValuePairs **=** **new** ArrayList**<**NameValuePair**>();**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"username"**,** username**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"password"**,** password**));**    httpPost**.**setEntity**(new** UrlEncodedFormEntity**(**nameValuePairs**));**  String authtoken **=** **null;**  **try** **{**  HttpResponse response **=** httpClient**.**execute**(**httpPost**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  CredentialsResponse registerResponse **=** **new** Gson**().**fromJson**(**responseString**,** CredentialsResponse**.**class**);**  **if** **(**registerResponse**.**statusCode **!=** 0**)** **{**  **throw** **new** Exception**(**"Error signing-in ["**+**registerResponse**.**statusCode**+**"] - " **+** registerResponse**.**statusMessage**);**  **}** **else** **{**  authtoken **=** registerResponse**.**secretToken**;**  **}**  Log**.**d**(**"Task"**,** "Response: "**+**registerResponse**.**toString**());**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **return** authtoken**;**  **}**  private class CredentialsResponse **implements** Serializable **{**    private static final long serialVersionUID **=** 6238763861656226544L**;**    int statusCode**;**  String statusMessage**;**  String secretToken**;**  @Override  public String toString**()** **{**  **return** "CredentialsResponse [statusCode=" **+** statusCode  **+** ", statusMessage=" **+** statusMessage **+** ", secretToken="  **+** secretToken **+** "]"**;**  **}**  **}**    **}** |

Task.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**db**;**  **import** java**.**util**.**Date**;**  **import** android**.**content**.**ContentValues**;**  **import** android**.**database**.**Cursor**;**  public class Task **{**  private int id**;**  private String title**;**  private long deadline**;**  private boolean done**;**  private boolean deleted **=** **false;**  private String remoteId**;**    public Task**(**int id**,** String title**,** long deadline**,** boolean done**,** boolean deleted**,** String remoteId**)** **{**  **this.**id **=** id**;**  **this.**title **=** title**;**  **this.**deadline **=** deadline**;**  **this.**done **=** done**;**  **this.**deleted **=** deleted**;**  **this.**remoteId **=** remoteId**;**  **}**  public Task**(**String title**,** long deadline**,** boolean done**,** boolean deleted**,** String remoteId**)** **{**  **this.**title **=** title**;**  **this.**deadline **=** deadline**;**  **this.**done **=** done**;**  **this.**deleted **=** deleted**;**  **this.**remoteId **=** remoteId**;**  **}**  public Task**()** **{**    **}**  public int getId**()** **{**  **return** id**;**  **}**  public void setId**(**int id**)** **{**  **this.**id **=** id**;**  **}**  public String getTitle**()** **{**  **return** title**;**  **}**  public void setTitle**(**String title**)** **{**  **this.**title **=** title**;**  **}**  public long getDeadline**()** **{**  **return** deadline**;**  **}**  public void setDeadline**(**long deadline**)** **{**  **this.**deadline **=** deadline**;**  **}**  public boolean isDone**()** **{**  **return** done**;**  **}**  public void setDone**(**boolean done**)** **{**  **this.**done **=** done**;**  **}**  public boolean isDeleted**()** **{**  **return** deleted**;**  **}**  public void setDeleted**(**boolean deleted**)** **{**  **this.**deleted **=** deleted**;**  **}**  public String getRemoteId**()** **{**  **return** remoteId**;**  **}**  public void setRemoteId**(**String remoteId**)** **{**  **this.**remoteId **=** remoteId**;**  **}**  @Override  public String toString**()** **{**  **return** "Task [id=" **+** id **+** ", title=" **+** title **+** ", deadline=" **+** deadline  **+** ", done=" **+** done **+** ", deleted=" **+** deleted **+** ", remoteId="  **+** remoteId **+** "]"**;**  **}**  public ContentValues getContentValues**()** **{**  ContentValues values **=** **new** ContentValues**();**  values**.**put**(**TaskSQLiteHelper**.**KEY\_ID**,** getId**());**  values**.**put**(**TaskSQLiteHelper**.**KEY\_TITLE**,** getTitle**());**  values**.**put**(**TaskSQLiteHelper**.**KEY\_DEADLINE**,** getDeadline**());**  values**.**put**(**TaskSQLiteHelper**.**KEY\_DONE**,** isDone**());**  values**.**put**(**TaskSQLiteHelper**.**KEY\_DELETED**,** isDeleted**());**  values**.**put**(**TaskSQLiteHelper**.**KEY\_REMOTE\_ID**,** getRemoteId**());**  **return** values**;**  **}**  public static Task fromCursor**(**Cursor curTvShows**)** **{**  int id **=** curTvShows**.**getInt**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_ID**));**  String title **=** curTvShows**.**getString**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_TITLE**));**  long deadline **=** curTvShows**.**getLong**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_DEADLINE**));**  boolean done **=** curTvShows**.**getInt**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_DONE**))** **==** 1 **?** **true** **:** **false;**  boolean deleted **=** curTvShows**.**getInt**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_DELETED**))** **==** 1 **?** **true** **:** **false;**  String remoteId **=** curTvShows**.**getString**(**curTvShows**.**getColumnIndex**(**TaskSQLiteHelper**.**KEY\_REMOTE\_ID**));**  **return** **new** Task**(**id**,** title**,** deadline**,** done**,** deleted**,** remoteId**);**  **}**    public id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**Task getTaskRemote**(){**  **return** **new** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**Task**(**title**,** **new** Date**(**deadline**),** done**,** deleted**);**  **}**  **}** |

TaskContentProvider.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**db**;**  **import** android**.**content**.**ContentProvider**;**  **import** android**.**content**.**ContentUris**;**  **import** android**.**content**.**ContentValues**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**UriMatcher**;**  **import** android**.**database**.**Cursor**;**  **import** android**.**database**.**sqlite**.**SQLiteDatabase**;**  **import** android**.**database**.**sqlite**.**SQLiteQueryBuilder**;**  **import** android**.**net**.**Uri**;**  **import** android**.**text**.**TextUtils**;**  public class TaskContentProvider **extends** ContentProvider **{**    public static final UriMatcher URI\_MATCHER **=** buildUriMatcher**();**  public static final String PATH **=** "task"**;**  public static final int PATH\_TOKEN **=** 100**;**  public static final String PATH\_FOR\_ID **=** "task/\*"**;**  public static final int PATH\_FOR\_ID\_TOKEN **=** 200**;**    public static final String AUTHORITY **=** "id.ac.gunadarma.tugasku.provider"**;**  public static final String CONTENT\_ITEM\_TYPE **=** "vnd.android.cursor.item/vnd.tugasku.task"**;**  public static final String CONTENT\_TYPE\_DIR **=** "vnd.android.cursor.dir/vnd.tugasku.task"**;**  public static final Uri CONTENT\_URI **=** Uri**.**parse**(**"content://"**+**AUTHORITY**+**"/task"**);**    private TaskSQLiteHelper dbHelper**;**    private static UriMatcher buildUriMatcher**()** **{**  final UriMatcher matcher **=** **new** UriMatcher**(**UriMatcher**.**NO\_MATCH**);**  final String authority **=** AUTHORITY**;**  matcher**.**addURI**(**authority**,** PATH**,** PATH\_TOKEN**);**  matcher**.**addURI**(**authority**,** PATH\_FOR\_ID**,** PATH\_FOR\_ID\_TOKEN**);**  **return** matcher**;**  **}**    @Override  public boolean onCreate**()** **{**  Context ctx **=** getContext**();**  dbHelper **=** **new** TaskSQLiteHelper**(**ctx**);**  **return** **true;**  **}**  @Override  public String getType**(**Uri uri**)** **{**  final int match **=** URI\_MATCHER**.**match**(**uri**);**  **switch** **(**match**)** **{**  **case** PATH\_TOKEN**:**  **return** CONTENT\_TYPE\_DIR**;**  **case** PATH\_FOR\_ID\_TOKEN**:**  **return** CONTENT\_ITEM\_TYPE**;**  **default:**  **throw** **new** UnsupportedOperationException**(**"URI " **+** uri **+** " is not supported."**);**  **}**  **}**  @Override  public Cursor query**(**Uri uri**,** String**[]** projection**,** String selection**,**  String**[]** selectionArgs**,** String sortOrder**)** **{**  SQLiteDatabase db **=** dbHelper**.**getReadableDatabase**();**  final int match **=** URI\_MATCHER**.**match**(**uri**);**  **switch** **(**match**)** **{**  **case** PATH\_TOKEN**:** **{**  SQLiteQueryBuilder builder **=** **new** SQLiteQueryBuilder**();**  builder**.**setTables**(**TaskSQLiteHelper**.**TABLE\_TASK**);**  **return** builder**.**query**(**db**,** projection**,** selection**,** selectionArgs**,** **null,** **null,** sortOrder**);**  **}**  **case** PATH\_FOR\_ID\_TOKEN**:** **{**  int tvShowId **=** **(**int**)**ContentUris**.**parseId**(**uri**);**  SQLiteQueryBuilder builder **=** **new** SQLiteQueryBuilder**();**  builder**.**setTables**(**TaskSQLiteHelper**.**TABLE\_TASK**);**  builder**.**appendWhere**(**TaskSQLiteHelper**.**KEY\_ID **+** "=" **+** tvShowId**);**  **return** builder**.**query**(**db**,** projection**,** selection**,**selectionArgs**,** **null,** **null,** sortOrder**);**  **}**  **default:**  **return** **null;**  **}**  **}**  @Override  public Uri insert**(**Uri uri**,** ContentValues values**)** **{**  SQLiteDatabase db **=** dbHelper**.**getWritableDatabase**();**  int token **=** URI\_MATCHER**.**match**(**uri**);**  **switch** **(**token**)** **{**  **case** PATH\_TOKEN**:** **{**  long id **=** db**.**insert**(**TaskSQLiteHelper**.**TABLE\_TASK**,** **null,** values**);**  **if** **(**id **!=** **-**1**)**  getContext**().**getContentResolver**().**notifyChange**(**uri**,** **null);**  **return** CONTENT\_URI**.**buildUpon**().**appendPath**(**String**.**valueOf**(**id**)).**build**();**  **}**  **default:** **{**  **throw** **new** UnsupportedOperationException**(**"URI: " **+** uri **+** " not supported."**);**  **}**  **}**  **}**  @Override  public int delete**(**Uri uri**,** String selection**,** String**[]** selectionArgs**)** **{**  SQLiteDatabase db **=** dbHelper**.**getWritableDatabase**();**  int token **=** URI\_MATCHER**.**match**(**uri**);**  int rowsDeleted **=** **-**1**;**  **switch** **(**token**)** **{**  **case** **(**PATH\_TOKEN**):**  rowsDeleted **=** db**.**delete**(**TaskSQLiteHelper**.**TABLE\_TASK**,** selection**,** selectionArgs**);**  **break;**  **case** **(**PATH\_FOR\_ID\_TOKEN**):**  String tvShowIdWhereClause **=** TaskSQLiteHelper**.**KEY\_ID **+** "=" **+** uri**.**getLastPathSegment**();**  **if** **(!**TextUtils**.**isEmpty**(**selection**))**  tvShowIdWhereClause **+=** " AND " **+** selection**;**  rowsDeleted **=** db**.**delete**(**TaskSQLiteHelper**.**TABLE\_TASK**,** tvShowIdWhereClause**,** selectionArgs**);**  **break;**  **default:**  **throw** **new** IllegalArgumentException**(**"Unsupported URI: " **+** uri**);**  **}**  **if** **(**rowsDeleted **!=** **-**1**)**  getContext**().**getContentResolver**().**notifyChange**(**uri**,** **null);**  **return** rowsDeleted**;**  **}**  @Override  public int update**(**Uri uri**,** ContentValues values**,** String selection**,** String**[]** selectionArgs**)** **{**  SQLiteDatabase db **=** dbHelper**.**getWritableDatabase**();**  **switch** **(**URI\_MATCHER**.**match**(**uri**))** **{**  **case** PATH\_TOKEN**:**  **break;**  **case** PATH\_FOR\_ID\_TOKEN**:**  String id **=** uri**.**getPathSegments**().**get**(**1**);**  selection **=** TaskSQLiteHelper**.**KEY\_ID  **+** "="  **+** id  **+** **(!**TextUtils**.**isEmpty**(**selection**)** **?** " AND (" **+** selection **+** ')' **:** ""**);**  **break;**  **default:**  **throw** **new** IllegalArgumentException**(**"Unsupported URI: " **+** uri**);**  **}**  int updateCount **=** db**.**update**(**TaskSQLiteHelper**.**TABLE\_TASK**,** values**,** selection**,** selectionArgs**);**  getContext**().**getContentResolver**().**notifyChange**(**uri**,** **null);**  **return** updateCount**;**  **}**  **}** |

TaskSQLiteHelper.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**db**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**Util**;**  **import** java**.**util**.**Calendar**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**LinkedList**;**  **import** java**.**util**.**List**;**  **import** android**.**content**.**ContentValues**;**  **import** android**.**content**.**Context**;**  **import** android**.**database**.**Cursor**;**  **import** android**.**database**.**sqlite**.**SQLiteDatabase**;**  **import** android**.**database**.**sqlite**.**SQLiteOpenHelper**;**  **import** android**.**util**.**Log**;**  public class TaskSQLiteHelper **extends** SQLiteOpenHelper **{**    private static final int DATABASE\_VERSION **=** 1**;**  public static final String DATABASE\_NAME **=** "TaskDB"**;**    public static final String TABLE\_TASK **=** "task"**;**  public static final String KEY\_ID **=** "id"**;**  public static final String KEY\_TITLE **=** "title"**;**  public static final String KEY\_DEADLINE **=** "deadline"**;**  public static final String KEY\_DONE **=** "done"**;**  public static final String KEY\_DELETED **=** "deleted"**;**  public static final String KEY\_REMOTE\_ID **=** "remote"**;**    public TaskSQLiteHelper**(**Context context**)** **{**  **super(**context**,** DATABASE\_NAME**,** **null,** DATABASE\_VERSION**);**  **}**  @Override  public void onCreate**(**SQLiteDatabase db**)** **{**  String CREATE\_TASK\_TABLE **=**  "CREATE TABLE "**+**TABLE\_TASK**+**" ( "  **+**KEY\_ID**+** " INTEGER PRIMARY KEY AUTOINCREMENT, "  **+**KEY\_TITLE**+** " TEXT, "  **+**KEY\_DEADLINE**+** " INTEGER, "  **+**KEY\_DONE**+** " INTEGER,"  **+**KEY\_DELETED**+** " INTEGER,"  **+**KEY\_REMOTE\_ID**+** " TEXT)"**;**  db**.**execSQL**(**CREATE\_TASK\_TABLE**);**  **}**  @Override  public void onUpgrade**(**SQLiteDatabase db**,** int oldVersion**,** int newVersion**)** **{**  db**.**execSQL**(**"DROP TABLE IF EXISTS "**+**TABLE\_TASK**);**  **this.**onCreate**(**db**);**  **}**    public void addTask**(**Task task**){**  Log**.**d**(**"addTask"**,** task**.**toString**());**  SQLiteDatabase db **=** **this.**getWritableDatabase**();**  ContentValues values **=** **new** ContentValues**();**  values**.**put**(**KEY\_TITLE**,** task**.**getTitle**());**  Calendar to **=** Calendar**.**getInstance**();**  to**.**setTime**(new** Date**(**task**.**getDeadline**()));**  values**.**put**(**KEY\_DEADLINE**,** to**.**getTimeInMillis**());**  values**.**put**(**KEY\_DONE**,** task**.**isDone**()** **?** 1 **:** 0**);**  values**.**put**(**KEY\_DELETED**,** task**.**isDeleted**()** **?** 1 **:** 0**);**  values**.**put**(**KEY\_REMOTE\_ID**,** task**.**getRemoteId**());**  db**.**insert**(**TABLE\_TASK**,** **null,** values**);**  db**.**close**();**  **}**    public void markTaskDelete**(**int id**){**  SQLiteDatabase db **=** **this.**getWritableDatabase**();**  ContentValues values **=** **new** ContentValues**();**  values**.**put**(**KEY\_DELETED**,** 1**);**  int i **=** db**.**update**(**TABLE\_TASK**,** values**,** KEY\_ID **+** " = ?"**,**  **new** String**[]** **{** String**.**valueOf**(**id**)** **});**  Log**.**d**(**"markTaskDelete("**+**id**+**")"**,** i**+**""**);**  db**.**close**();**  **}**    public void markTaskStatus**(**int id**,** boolean done**){**  SQLiteDatabase db **=** **this.**getWritableDatabase**();**  ContentValues values **=** **new** ContentValues**();**  values**.**put**(**KEY\_DONE**,** done **?** 1 **:** 0**);**  int i **=** db**.**update**(**TABLE\_TASK**,** values**,** KEY\_ID **+** " = ?"**,**  **new** String**[]** **{** String**.**valueOf**(**id**)** **});**  Log**.**d**(**"markTaskStatus("**+**id**+**")"**,** i**+**""**);**  db**.**close**();**  **}**    public void markSynced**(**int id**,** String remoteid**){**  SQLiteDatabase db **=** **this.**getWritableDatabase**();**  ContentValues values **=** **new** ContentValues**();**  values**.**put**(**KEY\_REMOTE\_ID**,** remoteid**);**  int i **=** db**.**update**(**TABLE\_TASK**,** values**,** KEY\_ID **+** " = ?"**,**  **new** String**[]** **{** String**.**valueOf**(**id**)** **});**  Log**.**d**(**"markSynced("**+**id**+**") ["**+**remoteid**+**"]"**,** i**+**""**);**  db**.**close**();**  **}**    public boolean isSynced**(**String remoteId**){**  String query **=** "SELECT \* FROM "**+**TABLE\_TASK**+**  " WHERE "**+**KEY\_REMOTE\_ID**+**" = '"**+**remoteId**+**"'"**;**  SQLiteDatabase db **=** **this.**getReadableDatabase**();**  Cursor cursor **=** db**.**rawQuery**(**query**,** **null);**  Task task **=** **null;**  **if** **(**cursor**.**moveToFirst**())** **{**  **do** **{**  task **=** **new** Task**();**  task**.**setId**(**cursor**.**getInt**(**0**));**  task**.**setTitle**(**cursor**.**getString**(**1**));**  task**.**setDeadline**(**cursor**.**getLong**(**2**));**  task**.**setDone**(**cursor**.**getInt**(**3**)** **==** 1 **?** **true** **:** **false);**  task**.**setDeleted**(**cursor**.**getInt**(**4**)** **==** 1 **?** **true** **:** **false);**  **}** **while** **(**cursor**.**moveToNext**());**  **}**  **return** task **!=** **null;**  **}**    public int getNearDeadline**(){**  String query **=** "SELECT \* FROM "**+**TABLE\_TASK**+**  " WHERE "**+**KEY\_DONE**+**" = '0'"**+**  " AND "**+**KEY\_DELETED**+**" = '0' ORDER BY "**+**KEY\_DEADLINE**+**" ASC LIMIT 1"**;**  SQLiteDatabase db **=** **this.**getReadableDatabase**();**  Cursor cursor **=** db**.**rawQuery**(**query**,** **null);**  int day **=** 0**;**  **if** **(**cursor**.**moveToFirst**())** **{**  **do** **{**  Calendar cal **=** Calendar**.**getInstance**();**  cal**.**setTime**(new** Date**(**cursor**.**getLong**(**2**)));**  day **=** Util**.**countDays**(**cal**);**  **}** **while** **(**cursor**.**moveToNext**());**  **}**  Log**.**d**(**"getNearDeadline()"**,** day**+**" days"**);**  db**.**close**();**  **return** day**;**  **}**    public List**<**Task**>** getTasks**(**boolean isDone**){**  List**<**Task**>** tasks **=** **new** LinkedList**<**Task**>();**  String query **=** "SELECT \* FROM "**+**TABLE\_TASK**+**  " WHERE "**+**KEY\_DONE**+**" = '"**+(**isDone**?** 1 **:** 0**)+**"'"**+**  " AND "**+**KEY\_DELETED**+**" = '0' ORDER BY "**+**KEY\_DEADLINE**+**" ASC"**;**  SQLiteDatabase db **=** **this.**getReadableDatabase**();**  Cursor cursor **=** db**.**rawQuery**(**query**,** **null);**  Task task **=** **null;**  **if** **(**cursor**.**moveToFirst**())** **{**  **do** **{**  task **=** **new** Task**();**  task**.**setId**(**cursor**.**getInt**(**0**));**  task**.**setTitle**(**cursor**.**getString**(**1**));**  task**.**setDeadline**(**cursor**.**getLong**(**2**));**  task**.**setDone**(**cursor**.**getInt**(**3**)** **==** 1 **?** **true** **:** **false);**  task**.**setDeleted**(**cursor**.**getInt**(**4**)** **==** 1 **?** **true** **:** **false);**  task**.**setRemoteId**(**cursor**.**getString**(**5**));**  tasks**.**add**(**task**);**  **}** **while** **(**cursor**.**moveToNext**());**  **}**  Log**.**d**(**"getTasks()"**,** tasks**.**toString**());**  db**.**close**();**  **return** tasks**;**  **}**  **}** |

MemoAlarmReceiver.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**.**alarm**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**MainActivity**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**R**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**TaskSQLiteHelper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**TimePreference**;**  **import** java**.**util**.**Calendar**;**  **import** android**.**app**.**AlarmManager**;**  **import** android**.**app**.**NotificationManager**;**  **import** android**.**app**.**PendingIntent**;**  **import** android**.**content**.**ComponentName**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**Intent**;**  **import** android**.**content**.**SharedPreferences**;**  **import** android**.**content**.**pm**.**PackageManager**;**  **import** android**.**preference**.**PreferenceManager**;**  **import** android**.**support**.**v4**.**app**.**NotificationCompat**;**  **import** android**.**support**.**v4**.**content**.**WakefulBroadcastReceiver**;**  public class MemoAlarmReceiver **extends** WakefulBroadcastReceiver **{**  private NotificationManager mNotificationManager**;**  NotificationCompat**.**Builder builder**;**  public static final int NOTIFICATION\_ID **=** 1**;**  private AlarmManager alarmMgr**;**  private PendingIntent alarmIntent**;**    @Override  public void onReceive**(**Context ctx**,** Intent intent**)** **{**  mNotificationManager **=** **(**NotificationManager**)** ctx**.**getSystemService**(**Context**.**NOTIFICATION\_SERVICE**);**  PendingIntent contentIntent **=** PendingIntent**.**getActivity**(**ctx**,** 0**,** **new** Intent**(**ctx**,** MainActivity**.**class**),** 0**);**  TaskSQLiteHelper sqLiteHelper **=** **new** TaskSQLiteHelper**(**ctx**);**  NotificationCompat**.**Builder mBuilder **=** **new** NotificationCompat**.**Builder**(**ctx**)**  **.**setAutoCancel**(true)**  **.**setSmallIcon**(**R**.**drawable**.**ic\_launcher**)**  **.**setContentTitle**(**"Pengingat Tugas"**)**  **.**setContentText**(**"Kamu mempunyai tugas penting dalam "**+**sqLiteHelper**.**getNearDeadline**()+**" hari"**);**  mBuilder**.**setContentIntent**(**contentIntent**);**  mNotificationManager**.**notify**(**NOTIFICATION\_ID**,** mBuilder**.**build**());**  **}**    public void setAlarm**(**Context context**)** **{**  alarmMgr **=** **(**AlarmManager**)**context**.**getSystemService**(**Context**.**ALARM\_SERVICE**);**  Intent intent **=** **new** Intent**(**context**,** MemoAlarmReceiver**.**class**);**  alarmIntent **=** PendingIntent**.**getBroadcast**(**context**,** 0**,** intent**,** 0**);**  SharedPreferences prefs **=** PreferenceManager**.**getDefaultSharedPreferences**(**context**);**  String time **=** prefs**.**getString**(**"notif\_time"**,** "08:00"**);**  System**.**out**.**println**(**"Set Next Alarm "**+**time**);**  int hour **=** TimePreference**.**getHour**(**time**);**  int minute **=** TimePreference**.**getMinute**(**time**);**    Calendar calendar **=** Calendar**.**getInstance**();**  calendar**.**setTimeInMillis**(**System**.**currentTimeMillis**());**  calendar**.**set**(**Calendar**.**HOUR\_OF\_DAY**,** hour**);**  calendar**.**set**(**Calendar**.**MINUTE**,** minute**);**    alarmMgr**.**cancel**(**alarmIntent**);**  alarmMgr**.**setInexactRepeating**(**AlarmManager**.**RTC\_WAKEUP**,**  calendar**.**getTimeInMillis**()** **+** AlarmManager**.**INTERVAL\_DAY**,** AlarmManager**.**INTERVAL\_DAY**,** alarmIntent**);**    ComponentName receiver **=** **new** ComponentName**(**context**,** MemoBootReceiver**.**class**);**  PackageManager pm **=** context**.**getPackageManager**();**  pm**.**setComponentEnabledSetting**(**receiver**,** PackageManager**.**COMPONENT\_ENABLED\_STATE\_ENABLED**,**  PackageManager**.**DONT\_KILL\_APP**);**  **}**  **}** |

MemoBootReceiver.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**.**alarm**;**  **import** android**.**content**.**BroadcastReceiver**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**Intent**;**  public class MemoBootReceiver **extends** BroadcastReceiver **{**  MemoAlarmReceiver alarm **=** **new** MemoAlarmReceiver**();**  @Override  public void onReceive**(**Context context**,** Intent intent**)** **{**  **if** **(**intent**.**getAction**().**equals**(**"android.intent.action.BOOT\_COMPLETED"**))** **{**  alarm**.**setAlarm**(**context**);**  **}**  **}**  **}** |

Task.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**;**  **import** java**.**util**.**Date**;**  **import** com**.**google**.**gson**.**annotations**.**Expose**;**  public class Task **{**  @Expose public String id**;**  @Expose public Date created**;**  @Expose public String content**;**  public String user**;**  @Expose public Date deadline**;**  @Expose public boolean done**;**  @Expose public boolean deleted**;**    public Task**()** **{}**  public Task**(**String content**,** Date deadline**,** boolean done**,** boolean deleted**)** **{**  **this.**content **=** content**;**  **this.**deadline **=** deadline**;**  **this.**done **=** done**;**  **this.**deleted **=** deleted**;**  **}**  public Task**(**String id**,** Date created**,** String content**,** String user**,** Date deadline**,** boolean done**,** boolean deleted**)** **{**  **this.**id **=** id**;**  **this.**created **=** created**;**  **this.**content **=** content**;**  **this.**user **=** user**;**  **this.**deadline **=** deadline**;**  **this.**done **=** done**;**  **this.**deleted **=** deleted**;**  **}**    public id**.**ac**.**gunadarma**.**tugasku**.**db**.**Task getTaskLocal**(){**  **return** **new** id**.**ac**.**gunadarma**.**tugasku**.**db**.**Task**(**content**,** deadline**.**getTime**(),** done**,** deleted**,** id**);**  **}**  **}** |

TaskResponse.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**;**  public class TaskResponse **{**  public String id**;**  public String statusCode**;**  public String statusMessage**;**  @Override  public String toString**()** **{**  **return** "TaskResponse [id=" **+** id **+** ", statusCode=" **+** statusCode  **+** ", statusMessage=" **+** statusMessage **+** "]"**;**  **}**    **}** |

Api.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**TaskResponse**;**  **import** java**.**io**.**IOException**;**  **import** java**.**lang**.**reflect**.**Type**;**  **import** java**.**text**.**SimpleDateFormat**;**  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**Date**;**  **import** java**.**util**.**List**;**  **import** org**.**apache**.**http**.**HttpResponse**;**  **import** org**.**apache**.**http**.**NameValuePair**;**  **import** org**.**apache**.**http**.**client**.**entity**.**UrlEncodedFormEntity**;**  **import** org**.**apache**.**http**.**client**.**methods**.**HttpGet**;**  **import** org**.**apache**.**http**.**client**.**methods**.**HttpPost**;**  **import** org**.**apache**.**http**.**impl**.**client**.**DefaultHttpClient**;**  **import** org**.**apache**.**http**.**message**.**BasicNameValuePair**;**  **import** org**.**apache**.**http**.**util**.**EntityUtils**;**  **import** android**.**util**.**Log**;**  **import** com**.**google**.**gson**.**Gson**;**  **import** com**.**google**.**gson**.**GsonBuilder**;**  **import** com**.**google**.**gson**.**JsonDeserializationContext**;**  **import** com**.**google**.**gson**.**JsonDeserializer**;**  **import** com**.**google**.**gson**.**JsonElement**;**  **import** com**.**google**.**gson**.**JsonParseException**;**  **import** com**.**google**.**gson**.**reflect**.**TypeToken**;**  public class Api **{**    public static final String BASE\_URL **=** "http://tugasku.herokuapp.com/"**;**  public static final String GET\_TASK **=** BASE\_URL **+** "/api/task/list"**;**  public static final String POST\_TASK **=** BASE\_URL **+** "/api/task/add"**;**  public static final String REMOVE\_TASK **=** BASE\_URL **+** "/api/task/remove"**;**  public static final String REGISTER **=** BASE\_URL **+** "/register"**;**    public static List**<**Task**>** getTaskList**(**String token**){**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  HttpGet httpGet **=** **new** HttpGet**(**GET\_TASK**);**  httpGet**.**addHeader**(**"Authorization"**,** token**);**  List**<**Task**>** taskList **=** **null;**  **try** **{**  HttpResponse response **=** httpClient**.**execute**(**httpGet**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  Log**.**d**(**"Api"**,** "Get Task ["**+**response**.**getStatusLine**()+**"] : "**+** responseString**);**  Type listType **=** **new** TypeToken**<**ArrayList**<**Task**>>()** **{}.**getType**();**  taskList **=** **new** GsonBuilder**()**  **.**excludeFieldsWithoutExposeAnnotation**()**  **.**registerTypeAdapter**(**Date**.**class**,** **new** JsonDeserializer**<**Date**>()** **{**  @Override  public Date deserialize**(**final JsonElement json**,** final Type typeOfT**,** final JsonDeserializationContext context**)** **throws** JsonParseException **{**  **return** **new** Date**(**json**.**getAsLong**());**  **}**  **}).**create**().**fromJson**(**responseString**,** listType**);**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **return** taskList**;**  **}**    public static void removeTask**(**String id**,** String token**){**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  HttpGet httpGet **=** **new** HttpGet**(**GET\_TASK**+**"?id="**+**id**);**  httpGet**.**addHeader**(**"Authorization"**,** token**);**  **try** **{**  HttpResponse response **=** httpClient**.**execute**(**httpGet**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  Log**.**d**(**"Api"**,** "Remove Task ["**+**response**.**getStatusLine**()+**"] ["**+**id**+**"] : "**+** responseString**);**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **}**    public static void register**(){**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  HttpPost httpPost **=** **new** HttpPost**(**REGISTER**);**  List**<**NameValuePair**>** nameValuePairs **=** **new** ArrayList**<**NameValuePair**>();**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"fname"**,** "kambing"**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"username"**,** "kambing"**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"password"**,** "kambing"**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"email"**,** "kambing@ui.ac.id"**));**  **try** **{**  httpPost**.**setEntity**(new** UrlEncodedFormEntity**(**nameValuePairs**));**  HttpResponse response **=** httpClient**.**execute**(**httpPost**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  Log**.**d**(**"Api"**,** "Register User ["**+**response**.**getStatusLine**()+**"] : "**+** responseString**);**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **}**    public static String postTask**(**Task task**,** String token**){**  DefaultHttpClient httpClient **=** **new** DefaultHttpClient**();**  HttpPost httpPost **=** **new** HttpPost**(**POST\_TASK**);**  httpPost**.**addHeader**(**"Authorization"**,** token**);**  List**<**NameValuePair**>** nameValuePairs **=** **new** ArrayList**<**NameValuePair**>();**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"content"**,** task**.**content**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"deadline"**,** **new** SimpleDateFormat**(**"yyyy-MM-dd"**).**format**(**task**.**deadline**)+**""**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"done"**,** task**.**done**+**""**));**  nameValuePairs**.**add**(new** BasicNameValuePair**(**"deleted"**,** task**.**deleted**+**""**));**  TaskResponse taskResponse**;**  **try** **{**  httpPost**.**setEntity**(new** UrlEncodedFormEntity**(**nameValuePairs**));**  HttpResponse response **=** httpClient**.**execute**(**httpPost**);**  String responseString **=** EntityUtils**.**toString**(**response**.**getEntity**());**  Log**.**d**(**"Api"**,** "Post Task ["**+**response**.**getStatusLine**()+**"] ["**+**task**.**content**+**"] : "**+** responseString**);**  taskResponse **=** **new** Gson**().**fromJson**(**responseString**,** TaskResponse**.**class**);**  **if** **(**response**.**getStatusLine**().**getStatusCode**()** **==** 200 **||** taskResponse**.**statusCode**.**equals**(**"00"**))** **{**  **return** taskResponse**.**id**;**  **}**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **return** ""**;**  **}**  **}** |

Util.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**helper**;**  **import** java**.**util**.**Calendar**;**  public class Util **{**  public static int countDays**(**Calendar to**)** **{**  Calendar from **=** Calendar**.**getInstance**();**  **return** diff**(**to**.**get**(**Calendar**.**YEAR**),** **(**to**.**get**(**Calendar**.**MONTH**)+**1**),** to**.**get**(**Calendar**.**DAY\_OF\_MONTH**),**  from**.**get**(**Calendar**.**YEAR**),** **(**from**.**get**(**Calendar**.**MONTH**)+**1**),** from**.**get**(**Calendar**.**DAY\_OF\_MONTH**));**  **}**  public static int julianDay**(**int year**,** int month**,** int day**)** **{**  int a **=** **(**14 **-** month**)** **/** 12**;**  int y **=** year **+** 4800 **-** a**;**  int m **=** month **+** 12 **\*** a **-** 3**;**  int jdn **=** day **+** **(**153 **\*** m **+** 2**)** **/** 5 **+** 365 **\*** y **+** y **/** 4 **-** y **/** 100 **+** y **/** 400  **-** 32045**;**  **return** jdn**;**  **}**  public static int diff**(**int y1**,** int m1**,** int d1**,** int y2**,** int m2**,** int d2**)** **{**  **return** julianDay**(**y1**,** m1**,** d1**)** **-** julianDay**(**y2**,** m2**,** d2**);**  **}**  **}** |

TaskSyncAdapter.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**sync**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**account**.**AccountGeneral**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**TaskContentProvider**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**TaskSQLiteHelper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**Api**;**  **import** java**.**io**.**IOException**;**  **import** java**.**util**.**ArrayList**;**  **import** java**.**util**.**List**;**  **import** android**.**accounts**.**Account**;**  **import** android**.**accounts**.**AccountManager**;**  **import** android**.**accounts**.**AuthenticatorException**;**  **import** android**.**accounts**.**OperationCanceledException**;**  **import** android**.**content**.**AbstractThreadedSyncAdapter**;**  **import** android**.**content**.**ContentProviderClient**;**  **import** android**.**content**.**ContentValues**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**SyncResult**;**  **import** android**.**database**.**Cursor**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**util**.**Log**;**  public class TaskSyncAdapter **extends** AbstractThreadedSyncAdapter **{**    private static final String TAG **=** "TaskSyncAdapter"**;**    private final AccountManager mAccountManager**;**  public TaskSyncAdapter**(**Context context**,** boolean autoInitialize**)** **{**  **super(**context**,** autoInitialize**);**  mAccountManager **=** AccountManager**.**get**(**context**);**  **}**    @Override  public void onPerformSync**(**Account account**,** Bundle extras**,** String authority**,**  ContentProviderClient provider**,** SyncResult syncResult**)** **{**  StringBuilder sb **=** **new** StringBuilder**();**  **if** **(**extras **!=** **null)** **{**  **for** **(**String key **:** extras**.**keySet**())** **{**  sb**.**append**(**key **+** "[" **+** extras**.**get**(**key**)** **+** "] "**);**  **}**  **}**  Log**.**d**(**TAG**,** "onPerformSync for account[" **+** account**.**name **+** "], Extras: " **+** sb**.**toString**());**    **try** **{**    String authToken **=** mAccountManager**.**blockingGetAuthToken**(**account**,** AccountManager**.**KEY\_AUTHTOKEN**,** **true);**  String userObjectId **=** mAccountManager**.**getUserData**(**account**,** AccountGeneral**.**USERDATA\_USER\_OBJ\_ID**);**    TaskSQLiteHelper sqLiteHelper **=** **new** TaskSQLiteHelper**(**getContext**());**    Log**.**d**(**TAG**,** "onPerformSync [" **+** authToken **+** "]. userObjectId: " **+** userObjectId**);**    List**<**id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**Task**>** remoteTaskList **=** Api**.**getTaskList**(**authToken**);**  Log**.**d**(**TAG**,** "onPerformSync, Remote Task: " **+** remoteTaskList**.**toString**());**    ArrayList**<**Task**>** localTaskList **=** **new** ArrayList**<**Task**>();**  Cursor curTvShows **=** provider**.**query**(**TaskContentProvider**.**CONTENT\_URI**,** **null,** **null,** **null,** **null);**  **if** **(**curTvShows **!=** **null)** **{**  **while** **(**curTvShows**.**moveToNext**())** **{**  localTaskList**.**add**(**Task**.**fromCursor**(**curTvShows**));**  **}**  curTvShows**.**close**();**  **}**  Log**.**d**(**TAG**,** "onPerformSync, Local Task: " **+** localTaskList**.**toString**());**    ArrayList**<**Task**>** taskToRemote **=** **new** ArrayList**<**Task**>();**  **for** **(**Task localTask **:** localTaskList**)** **{**  **if(**""**.**equals**(**localTask**.**getRemoteId**()))**  taskToRemote**.**add**(**localTask**);**  **}**  Log**.**d**(**TAG**,** "onPerformSync, Task to Remote: " **+** taskToRemote**.**toString**());**    ArrayList**<**Task**>** taskToLocal **=** **new** ArrayList**<**Task**>();**  **for** **(**id**.**ac**.**gunadarma**.**tugasku**.**helper**.**dao**.**Task remoteTask **:** remoteTaskList**)** **{**  **if(!**sqLiteHelper**.**isSynced**(**remoteTask**.**id**))**  taskToLocal**.**add**(**remoteTask**.**getTaskLocal**());**  **}**  Log**.**d**(**TAG**,** "onPerformSync, Task to Local: " **+** taskToLocal**.**toString**());**    **if** **(**taskToRemote**.**size**()** **==** 0**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> No local changes to update server"**);**  **}** **else** **{**  Log**.**d**(**"Task"**,** TAG **+** "> Updating remote server with local changes"**);**  **for** **(**Task remoteTask **:** taskToRemote**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> Local -> Remote [" **+** remoteTask**.**getId**()** **+** "]"**);**  String id **=** Api**.**postTask**(**remoteTask**.**getTaskRemote**(),** authToken**);**  **if(!**""**.**equals**(**id**)){**  sqLiteHelper**.**markSynced**(**remoteTask**.**getId**(),** id**);**  **}**  **}**  **}**    **if** **(**taskToLocal**.**size**()** **==** 0**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> No server changes to update local database"**);**  **}** **else** **{**  Log**.**d**(**"Task"**,** TAG **+** "> Updating local database with remote changes"**);**  int i **=** 0**;**  ContentValues showsToLocalValues**[]** **=** **new** ContentValues**[**taskToLocal**.**size**()];**  **for** **(**Task localTask **:** taskToLocal**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> Remote -> Local [" **+** localTask**.**getRemoteId**()** **+** "]"**);**  showsToLocalValues**[**i**++]** **=** localTask**.**getContentValues**();**  **}**  provider**.**bulkInsert**(**TaskContentProvider**.**CONTENT\_URI**,** showsToLocalValues**);**  **}**  Log**.**d**(**"Task"**,** TAG **+** "> Finished."**);**    **}** **catch** **(**OperationCanceledException e**)** **{**  e**.**printStackTrace**();**  **}** **catch** **(**IOException e**)** **{**  syncResult**.**stats**.**numIoExceptions**++;**  e**.**printStackTrace**();**  **}** **catch** **(**AuthenticatorException e**)** **{**  syncResult**.**stats**.**numAuthExceptions**++;**  e**.**printStackTrace**();**  **}** **catch** **(**Exception e**)** **{**  e**.**printStackTrace**();**  **}**  **}**  **}** |

TaskSyncService.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**.**sync**;**  **import** android**.**app**.**Service**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**IBinder**;**  public class TaskSyncService **extends** Service **{**  private static final Object sSyncAdapterLock **=** **new** Object**();**  private static TaskSyncAdapter sSyncAdapter **=** **null;**  @Override  public void onCreate**()** **{**  synchronized **(**sSyncAdapterLock**)** **{**  **if** **(**sSyncAdapter **==** **null)**  sSyncAdapter **=** **new** TaskSyncAdapter**(**getApplicationContext**(),** **true);**  **}**  **}**  @Override  public IBinder onBind**(**Intent intent**)** **{**  **return** sSyncAdapter**.**getSyncAdapterBinder**();**  **}**  **}** |

FragmentTaskPager.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**alarm**.**MemoAlarmReceiver**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**TaskAdapter**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**TaskListFragment**;**  **import** java**.**util**.**ArrayList**;**  **import** android**.**content**.**Context**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**support**.**v4**.**app**.**Fragment**;**  **import** android**.**support**.**v4**.**app**.**FragmentActivity**;**  **import** android**.**support**.**v4**.**app**.**FragmentPagerAdapter**;**  **import** android**.**support**.**v4**.**view**.**ViewPager**;**  **import** android**.**view**.**View**;**  **import** android**.**view**.**View**.**OnClickListener**;**  **import** android**.**view**.**ViewGroup**;**  **import** android**.**widget**.**TabHost**;**  **import** android**.**widget**.**TabWidget**;**  **import** com**.**actionbarsherlock**.**app**.**SherlockFragmentActivity**;**  public class FragmentTabsPager **extends** SherlockFragmentActivity **{**  TabHost mTabHost**;**  ViewPager mViewPager**;**  TabsAdapter mTabsAdapter**;**  @Override  protected void onCreate**(**Bundle savedInstanceState**)** **{**  **super.**onCreate**(**savedInstanceState**);**  setContentView**(**R**.**layout**.**fragment\_tabs\_pager**);**  getSupportActionBar**().**setDisplayShowCustomEnabled**(true);**  getSupportActionBar**().**setDisplayShowHomeEnabled**(false);**  getSupportActionBar**().**setCustomView**(**R**.**layout**.**actionbar**);**    mTabHost **=** **(**TabHost**)** findViewById**(**android**.**R**.**id**.**tabhost**);**  mTabHost**.**setup**();**  mViewPager **=** **(**ViewPager**)** findViewById**(**R**.**id**.**pager**);**  mTabsAdapter **=** **new** TabsAdapter**(this,** mTabHost**,** mViewPager**);**  Bundle args **=** **new** Bundle**();**  args**.**putBoolean**(**"task"**,** **true);**  mTabsAdapter**.**addTab**(**mTabHost**.**newTabSpec**(**"task"**).**setIndicator**(**"Task"**),**  TaskListFragment**.**class**,** args**);**  args **=** **new** Bundle**();**  args**.**putBoolean**(**"task"**,** **false);**  mTabsAdapter**.**addTab**(**  mTabHost**.**newTabSpec**(**"history"**).**setIndicator**(**"History"**),**  TaskListFragment**.**class**,** args**);**  **if** **(**savedInstanceState **!=** **null)** **{**  mTabHost**.**setCurrentTabByTag**(**savedInstanceState**.**getString**(**"tab"**));**  **}**    getActionBar**().**getCustomView**().**findViewById**(**R**.**id**.**imageButton2**).**setOnClickListener**(new** OnClickListener**()** **{**    @Override  public void onClick**(**View arg0**)** **{**  startActivityForResult**(new** Intent**(**FragmentTabsPager**.this,** TaskFormFragmentActivity**.**class**),** 1**);**  **}**  **});**    getActionBar**().**getCustomView**().**findViewById**(**R**.**id**.**imageButton1**).**setOnClickListener**(new** OnClickListener**()** **{**    @Override  public void onClick**(**View arg0**)** **{**  Intent intent **=** **new** Intent**(**FragmentTabsPager**.this,** PreferenceActivity**.**class**);**  startActivityForResult**(**intent**,** 0**);**  **}**  **});**  **}**  public void reloadList**(**int page**)** **{**  TaskListFragment listFragment **=** **(**TaskListFragment**)** getSupportFragmentManager**()**  **.**getFragments**().**get**(**page**);**  TaskAdapter adapter **=** **(**TaskAdapter**)** listFragment**.**getListAdapter**();**  listFragment**.**reload**(**page **==** 1 **?** **true** **:** **false);**  adapter**.**notifyDataSetChanged**();**  **}**  @Override  protected void onActivityResult**(**int requestCode**,** int resultCode**,** Intent data**)** **{**  **if** **(**requestCode **==** 1**)** **{**  **if** **(**resultCode **==** RESULT\_OK**)** **{**  reloadList**(**0**);**  **}**  **}** **else** **if** **(**requestCode **==** 0**)** **{**  **if** **(**resultCode **==** RESULT\_OK**)** **{**  **new** MemoAlarmReceiver**().**setAlarm**(this);**  **}**  **}**  **}**  // @Override  // public boolean onCreateOptionsMenu(Menu menu) {  // MenuInflater inflater = getSupportMenuInflater();  // inflater.inflate(R.menu.main, menu);  // return true;  // }  //  // @Override  // public boolean onOptionsItemSelected(MenuItem item) {  // switch (item.getItemId()) {  // case R.id.action\_add:  // startActivityForResult(new Intent(this,  // TaskFormFragmentActivity.class), 1);  // return true;  // case R.id.action\_preference:  // Intent intent = new Intent(this, PreferenceActivity.class);  // startActivityForResult(intent, 0);  // return true;  // default:  // return super.onOptionsItemSelected(item);  // }  // }  @Override  protected void onSaveInstanceState**(**Bundle outState**)** **{**  **super.**onSaveInstanceState**(**outState**);**  outState**.**putString**(**"tab"**,** mTabHost**.**getCurrentTabTag**());**  **}**  public static class TabsAdapter **extends** FragmentPagerAdapter **implements**  TabHost**.**OnTabChangeListener**,** ViewPager**.**OnPageChangeListener **{**  private final Context mContext**;**  private final TabHost mTabHost**;**  private final ViewPager mViewPager**;**  private final ArrayList**<**TabInfo**>** mTabs **=** **new** ArrayList**<**TabInfo**>();**  static final class TabInfo **{**  private final String tag**;**  private final Class**<?>** clss**;**  private final Bundle args**;**  TabInfo**(**String \_tag**,** Class**<?>** \_class**,** Bundle \_args**)** **{**  tag **=** \_tag**;**  clss **=** \_class**;**  args **=** \_args**;**  **}**  **}**  static class DummyTabFactory **implements** TabHost**.**TabContentFactory **{**  private final Context mContext**;**  public DummyTabFactory**(**Context context**)** **{**  mContext **=** context**;**  **}**  @Override  public View createTabContent**(**String tag**)** **{**  View v **=** **new** View**(**mContext**);**  v**.**setMinimumWidth**(**0**);**  v**.**setMinimumHeight**(**0**);**  **return** v**;**  **}**  **}**  public TabsAdapter**(**FragmentActivity activity**,** TabHost tabHost**,**  ViewPager pager**)** **{**  **super(**activity**.**getSupportFragmentManager**());**  mContext **=** activity**;**  mTabHost **=** tabHost**;**  mViewPager **=** pager**;**  mTabHost**.**setOnTabChangedListener**(this);**  mViewPager**.**setAdapter**(this);**  mViewPager**.**setOnPageChangeListener**(this);**  **}**  public void addTab**(**TabHost**.**TabSpec tabSpec**,** Class**<?>** clss**,** Bundle args**)** **{**  tabSpec**.**setContent**(new** DummyTabFactory**(**mContext**));**  String tag **=** tabSpec**.**getTag**();**  TabInfo info **=** **new** TabInfo**(**tag**,** clss**,** args**);**  mTabs**.**add**(**info**);**  mTabHost**.**addTab**(**tabSpec**);**  notifyDataSetChanged**();**  **}**  @Override  public int getCount**()** **{**  **return** mTabs**.**size**();**  **}**  @Override  public Fragment getItem**(**int position**)** **{**  TabInfo info **=** mTabs**.**get**(**position**);**  **return** Fragment**.**instantiate**(**mContext**,** info**.**clss**.**getName**(),**  info**.**args**);**  **}**  @Override  public void onTabChanged**(**String tabId**)** **{**  int position **=** mTabHost**.**getCurrentTab**();**  mViewPager**.**setCurrentItem**(**position**);**  **}**  @Override  public void onPageScrolled**(**int position**,** float positionOffset**,**  int positionOffsetPixels**)** **{**  **}**  @Override  public void onPageSelected**(**int position**)** **{**  TabWidget widget **=** mTabHost**.**getTabWidget**();**  int oldFocusability **=** widget**.**getDescendantFocusability**();**  widget**.**setDescendantFocusability**(**ViewGroup**.**FOCUS\_BLOCK\_DESCENDANTS**);**  mTabHost**.**setCurrentTab**(**position**);**  widget**.**setDescendantFocusability**(**oldFocusability**);**  **}**  @Override  public void onPageScrollStateChanged**(**int state**)** **{**  **}**  **}**  **}** |

LoginActivity.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**account**.**AccountGeneral**;**  **import** android**.**accounts**.**Account**;**  **import** android**.**accounts**.**AccountAuthenticatorActivity**;**  **import** android**.**accounts**.**AccountManager**;**  **import** android**.**app**.**ProgressDialog**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**AsyncTask**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**util**.**Log**;**  **import** android**.**view**.**View**;**  **import** android**.**widget**.**EditText**;**  **import** android**.**widget**.**Toast**;**  public class LoginActivity **extends** AccountAuthenticatorActivity **{**  public final static String ARG\_ACCOUNT\_TYPE **=** "ACCOUNT\_TYPE"**;**  public final static String ARG\_AUTH\_TYPE **=** "AUTH\_TYPE"**;**  public final static String ARG\_ACCOUNT\_NAME **=** "ACCOUNT\_NAME"**;**  public final static String ARG\_IS\_ADDING\_NEW\_ACCOUNT **=** "IS\_ADDING\_ACCOUNT"**;**  public static final String KEY\_ERROR\_MESSAGE **=** "ERR\_MSG"**;**  public final static String PARAM\_USER\_PASS **=** "USER\_PASS"**;**  private final int REQ\_SIGNUP **=** 1**;**  private final String TAG **=** **this.**getClass**().**getSimpleName**();**  private AccountManager mAccountManager**;**    private String mAuthTokenType **=** AccountGeneral**.**AUTHTOKEN\_TYPE\_FULL\_ACCESS**;**  /\*\*  \* Called when the activity is first created.  \*/  @Override  public void onCreate**(**Bundle savedInstanceState**)** **{**  **super.**onCreate**(**savedInstanceState**);**  setContentView**(**R**.**layout**.**login\_layout**);**  mAccountManager **=** AccountManager**.**get**(**getBaseContext**());**      findViewById**(**R**.**id**.**sign**).**setOnClickListener**(new** View**.**OnClickListener**()** **{**  @Override  public void onClick**(**View v**)** **{**  signIn**();**  **}**  **});**    /\*findViewById(R.id.signUp).setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Since there can only be one AuthenticatorActivity, we call the sign up activity, get his results,  // and return them in setAccountAuthenticatorResult(). See finishLogin().  Intent signup = new Intent(getBaseContext(), SignUpActivity.class);  signup.putExtras(getIntent().getExtras());  startActivityForResult(signup, REQ\_SIGNUP);  }  });\*/    **}**  @Override  protected void onActivityResult**(**int requestCode**,** int resultCode**,** Intent data**)** **{**  **if** **(**requestCode **==** REQ\_SIGNUP **&&** resultCode **==** RESULT\_OK**)** **{**  finishLogin**(**data**);**  **}** **else**  **super.**onActivityResult**(**requestCode**,** resultCode**,** data**);**  **}**  public void signIn**()** **{**  final String userName **=** **((**EditText**)** findViewById**(**R**.**id**.**username**)).**getText**().**toString**();**  final String userPass **=** **((**EditText**)** findViewById**(**R**.**id**.**password**)).**getText**().**toString**();**  **new** AsyncTask**<**String**,** Void**,** Intent**>()** **{**  ProgressDialog dialog**;**  protected void onPreExecute**()** **{**  dialog **=** ProgressDialog**.**show**(**LoginActivity**.this,** ""**,** "Loading. Please wait..."**,** **true);**  **};**  @Override  protected Intent doInBackground**(**String**...** params**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> Started authenticating"**);**  String authtoken **=** **null;**  Bundle data **=** **new** Bundle**();**  **try** **{**  authtoken **=** AccountGeneral**.**sServerAuthenticate**.**userSignIn**(**userName**,** userPass**,** mAuthTokenType**);**  data**.**putString**(**AccountManager**.**KEY\_ACCOUNT\_NAME**,** userName**);**  data**.**putString**(**AccountManager**.**KEY\_ACCOUNT\_TYPE**,** AccountGeneral**.**ACCOUNT\_TYPE**);**  data**.**putString**(**AccountManager**.**KEY\_AUTHTOKEN**,** authtoken**);**    Bundle userData **=** **new** Bundle**();**  userData**.**putString**(**AccountGeneral**.**USERDATA\_USER\_OBJ\_ID**,** "monyet"**);**  data**.**putBundle**(**AccountManager**.**KEY\_USERDATA**,** userData**);**    data**.**putString**(**PARAM\_USER\_PASS**,** userPass**);**  **}** **catch** **(**Exception e**)** **{**  data**.**putString**(**KEY\_ERROR\_MESSAGE**,** e**.**getMessage**());**  **}**  final Intent res **=** **new** Intent**();**  res**.**putExtras**(**data**);**  **return** res**;**  **}**  @Override  protected void onPostExecute**(**Intent intent**)** **{**  dialog**.**dismiss**();**  **if** **(**intent**.**hasExtra**(**KEY\_ERROR\_MESSAGE**))** **{**  Toast**.**makeText**(**getBaseContext**(),** intent**.**getStringExtra**(**KEY\_ERROR\_MESSAGE**),** Toast**.**LENGTH\_SHORT**).**show**();**  **}** **else** **{**  finishLogin**(**intent**);**  **}**  **}**  **}.**execute**();**  **}**  private void finishLogin**(**Intent intent**)** **{**  Log**.**d**(**"Task"**,** TAG **+** "> finishLogin"**);**  String accountName **=** intent**.**getStringExtra**(**AccountManager**.**KEY\_ACCOUNT\_NAME**);**  String accountPassword **=** intent**.**getStringExtra**(**PARAM\_USER\_PASS**);**  final Account account **=** **new** Account**(**accountName**,** intent**.**getStringExtra**(**AccountManager**.**KEY\_ACCOUNT\_TYPE**));**  **if** **(**getIntent**().**getBooleanExtra**(**ARG\_IS\_ADDING\_NEW\_ACCOUNT**,** **false))** **{**  Log**.**d**(**"Task"**,** TAG **+** "> finishLogin > addAccountExplicitly, AuthToken: "**+**intent**.**getStringExtra**(**AccountManager**.**KEY\_AUTHTOKEN**));**  String authtoken **=** intent**.**getStringExtra**(**AccountManager**.**KEY\_AUTHTOKEN**);**  String authtokenType **=** mAuthTokenType**;**  mAccountManager**.**addAccountExplicitly**(**account**,** accountPassword**,** intent**.**getBundleExtra**(**AccountManager**.**KEY\_USERDATA**));**  mAccountManager**.**setAuthToken**(**account**,** authtokenType**,** authtoken**);**  **}** **else** **{**  Log**.**d**(**"udinic"**,** TAG **+** "> finishLogin > setPassword"**);**  mAccountManager**.**setPassword**(**account**,** accountPassword**);**  **}**  setAccountAuthenticatorResult**(**intent**.**getExtras**());**  setResult**(**RESULT\_OK**,** intent**);**  finish**();**  **}**  **}** |

MainActivity.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**alarm**.**MemoAlarmReceiver**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**TaskAdapter**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**TaskListFragment**;**  **import** android**.**app**.**ActionBar**;**  **import** android**.**app**.**FragmentTransaction**;**  **import** android**.**content**.**Intent**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**support**.**v4**.**app**.**Fragment**;**  **import** android**.**support**.**v4**.**app**.**FragmentActivity**;**  **import** android**.**support**.**v4**.**app**.**FragmentManager**;**  **import** android**.**support**.**v4**.**app**.**FragmentPagerAdapter**;**  **import** android**.**support**.**v4**.**view**.**ViewPager**;**  **import** android**.**view**.**Menu**;**  **import** android**.**view**.**MenuInflater**;**  **import** android**.**view**.**MenuItem**;**  **import** android**.**widget**.**Toast**;**  public class MainActivity **extends** FragmentActivity **implements**  ActionBar**.**TabListener **{**  AppSectionsPagerAdapter mAppSectionsPagerAdapter**;**  ViewPager mViewPager**;**  public void onCreate**(**Bundle savedInstanceState**)** **{**  **super.**onCreate**(**savedInstanceState**);**  setContentView**(**R**.**layout**.**activity\_main**);**  **new** MemoAlarmReceiver**().**setAlarm**(this);**  mAppSectionsPagerAdapter **=** **new** AppSectionsPagerAdapter**(**  getSupportFragmentManager**());**  final ActionBar actionBar **=** getActionBar**();**  actionBar**.**setHomeButtonEnabled**(false);**  actionBar**.**setNavigationMode**(**ActionBar**.**NAVIGATION\_MODE\_TABS**);**  mViewPager **=** **(**ViewPager**)** findViewById**(**R**.**id**.**pager**);**  mViewPager**.**setAdapter**(**mAppSectionsPagerAdapter**);**  mViewPager  **.**setOnPageChangeListener**(new** ViewPager**.**SimpleOnPageChangeListener**()** **{**  @Override  public void onPageSelected**(**int position**)** **{**  actionBar**.**setSelectedNavigationItem**(**position**);**  **}**  **});**  **for** **(**int i **=** 0**;** i **<** mAppSectionsPagerAdapter**.**getCount**();** i**++)** **{**  actionBar**.**addTab**(**actionBar**.**newTab**()**  **.**setText**(**mAppSectionsPagerAdapter**.**getPageTitle**(**i**))**  **.**setTabListener**(this));**  **}**  **}**  @Override  public void onTabUnselected**(**ActionBar**.**Tab tab**,**  FragmentTransaction fragmentTransaction**)** **{**  **}**  @Override  public void onTabSelected**(**ActionBar**.**Tab tab**,**  FragmentTransaction fragmentTransaction**)** **{**  mViewPager**.**setCurrentItem**(**tab**.**getPosition**());**  **}**  @Override  public void onTabReselected**(**ActionBar**.**Tab tab**,**  FragmentTransaction fragmentTransaction**)** **{**  **}**  public static class AppSectionsPagerAdapter **extends** FragmentPagerAdapter **{**  public AppSectionsPagerAdapter**(**FragmentManager fm**)** **{**  **super(**fm**);**  **}**  @Override  public Fragment getItem**(**int i**)** **{**  TaskListFragment fragment **=** **new** TaskListFragment**();**  Bundle args **=** **new** Bundle**();**  **switch** **(**i**)** **{**  **case** 0**:**  args**.**putBoolean**(**"task"**,** **true);**  fragment**.**setArguments**(**args**);**  **break;**  **case** 1**:**  args**.**putBoolean**(**"task"**,** **false);**  fragment**.**setArguments**(**args**);**  **break;**  **}**  **return** fragment**;**  **}**  @Override  public int getCount**()** **{**  **return** 2**;**  **}**  @Override  public CharSequence getPageTitle**(**int position**)** **{**  **if** **(**position **==** 0**)** **{**  **return** "Task"**;**  **}** **else** **{**  **return** "History"**;**  **}**  **}**  **}**    public void reloadList**(**int page**){**  TaskListFragment listFragment **=** **(**TaskListFragment**)** getSupportFragmentManager**().**getFragments**().**get**(**page**);**  TaskAdapter adapter **=** **(**TaskAdapter**)** listFragment**.**getListAdapter**();**  listFragment**.**reload**(**page **==** 1 **?** **true** **:** **false);**  adapter**.**notifyDataSetChanged**();**  **}**    @Override  protected void onActivityResult**(**int requestCode**,** int resultCode**,** Intent data**)** **{**  **if** **(**requestCode **==** 1**)** **{**  **if** **(**resultCode **==** RESULT\_OK**)** **{**  reloadList**(**0**);**  **}**  **}else** **if** **(**requestCode **==** 0**)** **{**  **if** **(**resultCode **==** RESULT\_OK**)** **{**  **new** MemoAlarmReceiver**().**setAlarm**(this);**  **}**  **}**  **}**  @Override  public boolean onOptionsItemSelected**(**MenuItem item**)** **{**  **switch** **(**item**.**getItemId**())** **{**  **case** R**.**id**.**action\_add**:**  startActivityForResult**(new** Intent**(this,** TaskFormFragmentActivity**.**class**),** 1**);**  **return** **true;**  **case** R**.**id**.**action\_preference**:**  Intent intent **=** **new** Intent**(this,** PreferenceActivity**.**class**);**  startActivityForResult**(**intent**,** 0**);**  **return** **true;**  **default:**  **return** **super.**onOptionsItemSelected**(**item**);**  **}**  **}**  @Override  public boolean onCreateOptionsMenu**(**Menu menu**)** **{**  MenuInflater inflater **=** getMenuInflater**();**  inflater**.**inflate**(**R**.**menu**.**main**,** menu**);**  **return** **super.**onCreateOptionsMenu**(**menu**);**  **}**  **}** |

PreferenceActivity.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**PrefsFragment**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**support**.**v4**.**app**.**FragmentActivity**;**  public class PreferenceActivity **extends** FragmentActivity **{**  @Override  protected void onCreate**(**Bundle arg0**)** **{**  **super.**onCreate**(**arg0**);**  getFragmentManager**().**beginTransaction**()**  **.**replace**(**android**.**R**.**id**.**content**,** **new** PrefsFragment**()).**commit**();**  **}**  @Override  public void finish**()** **{**  setResult**(**RESULT\_OK**,** **null);**  **super.**finish**();**  **}**  **}** |

TaskFormFragmentActivity.java

|  |
| --- |
| package id**.**ac**.**gunadarma**.**tugasku**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**Task**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**db**.**TaskSQLiteHelper**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**helper**.**Util**;**  **import** id**.**ac**.**gunadarma**.**tugasku**.**ui**.**DatePickerFragment**;**  **import** java**.**util**.**Calendar**;**  **import** android**.**app**.**DatePickerDialog**.**OnDateSetListener**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**support**.**v4**.**app**.**DialogFragment**;**  **import** android**.**view**.**View**;**  **import** android**.**view**.**View**.**OnClickListener**;**  **import** android**.**widget**.**DatePicker**;**  **import** android**.**widget**.**EditText**;**  **import** android**.**widget**.**TextView**;**  **import** android**.**widget**.**Toast**;**  **import** com**.**actionbarsherlock**.**app**.**SherlockFragmentActivity**;**  public class TaskFormFragmentActivity **extends** SherlockFragmentActivity **implements** OnDateSetListener **{**  private TextView textView**;**  private EditText titleEt**;**  private long date**;**    @Override  protected void onCreate**(**Bundle savedInstanceState**)** **{**  **super.**onCreate**(**savedInstanceState**);**  setContentView**(**R**.**layout**.**taks\_form\_layout**);**    getSupportActionBar**().**setDisplayShowCustomEnabled**(true);**  getSupportActionBar**().**setDisplayShowHomeEnabled**(false);**  getSupportActionBar**().**setCustomView**(**R**.**layout**.**actionbarsave**);**    textView **=** **(**TextView**)** findViewById**(**R**.**id**.**textView1**);**  titleEt **=** **(**EditText**)** findViewById**(**R**.**id**.**editText1**);**    getActionBar**().**getCustomView**().**findViewById**(**R**.**id**.**imageButton1**).**setOnClickListener**(new** OnClickListener**()** **{**    @Override  public void onClick**(**View arg0**)** **{**  saveTask**();**  **}**  **});**  **}**  private void saveTask**()** **{**  TaskSQLiteHelper db **=** **new** TaskSQLiteHelper**(this);**  db**.**addTask**(new** Task**(**titleEt**.**getText**().**toString**(),** date**,** **false,** **false,** ""**));**  Toast**.**makeText**(this,** "Success"**,** Toast**.**LENGTH\_LONG**).**show**();**  setResult**(**RESULT\_OK**,** **null);**  finish**();**  **}**  public void showDatePickerDialog**(**View v**)** **{**  DialogFragment newFragment **=** **new** DatePickerFragment**();**  newFragment**.**show**(**getSupportFragmentManager**(),** "datePicker"**);**  **}**  @Override  public void onDateSet**(**DatePicker view**,** int year**,** int monthOfYear**,**int dayOfMonth**)** **{**  Calendar calendar **=** Calendar**.**getInstance**();**  calendar**.**set**(**Calendar**.**YEAR**,** year**);**  calendar**.**set**(**Calendar**.**MONTH**,** monthOfYear**);**  calendar**.**set**(**Calendar**.**DAY\_OF\_MONTH**,** dayOfMonth**);**  calendar**.**set**(**Calendar**.**HOUR**,** 0**);**  calendar**.**set**(**Calendar**.**MINUTE**,** 0**);**  calendar**.**set**(**Calendar**.**SECOND**,** 0**);**  textView**.**setText**(**dayOfMonth **+** "-" **+** **(**monthOfYear**+**1**)** **+** "-" **+** year**+**" ("**+**Util**.**countDays**(**calendar**)+**" Hari Lagi)"**);**  date **=** calendar**.**getTimeInMillis**();**  **}**  **}** |

actionbar.xml

|  |
| --- |
| <RelativeLayout xmlns:android=**"http://schemas.android.com/apk/res/android"**  android:layout\_width=**"fill\_parent"**  android:layout\_height=**"@dimen/abs\_\_action\_bar\_default\_height"**  android:background=**"#55ACEE"** >  <ImageButton  android:id=**"@+id/imageButton1"**  android:layout\_width=**"27dp"**  android:layout\_height=**"27dp"**  android:layout\_alignParentRight=**"true"**  android:layout\_centerVertical=**"true"**  android:layout\_marginRight=**"10dp"**  android:background=**"@null"**  android:scaleType=**"fitStart"**  android:src=**"@drawable/ic\_settings"** />  <ImageButton  android:id=**"@+id/imageButton2"**  android:layout\_width=**"27dp"**  android:layout\_height=**"27dp"**  android:layout\_centerVertical=**"true"**  android:layout\_marginRight=**"20dp"**  android:layout\_toLeftOf=**"@+id/imageButton1"**  android:src=**"@drawable/ic\_plus"**  android:scaleType=**"fitStart"**  android:background=**"@null"**/>  <ImageView  android:id=**"@+id/imageView1"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignParentLeft=**"true"**  android:layout\_centerVertical=**"true"**  android:layout\_marginLeft=**"10dp"**  android:src=**"@drawable/logo"** />  </RelativeLayout> |

actionbarsave.xml

|  |
| --- |
| <RelativeLayout xmlns:android=**"http://schemas.android.com/apk/res/android"**  android:layout\_width=**"fill\_parent"**  android:layout\_height=**"@dimen/abs\_\_action\_bar\_default\_height"**  android:background=**"#55ACEE"** >  <ImageButton  android:id=**"@+id/imageButton1"**  android:layout\_width=**"27dp"**  android:layout\_height=**"27dp"**  android:layout\_alignParentRight=**"true"**  android:layout\_centerVertical=**"true"**  android:layout\_marginRight=**"10dp"**  android:background=**"@null"**  android:scaleType=**"fitStart"**  android:src=**"@drawable/ic\_plus"** />  <ImageView  android:id=**"@+id/imageView1"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignParentLeft=**"true"**  android:layout\_centerVertical=**"true"**  android:layout\_marginLeft=**"10dp"**  android:src=**"@drawable/logo"** />  </RelativeLayout> |

activity\_main.xml

|  |
| --- |
| <!--  Copyright 2012 The Android Open Source Project  Licensed under the Apache License, Version 2.0 (the "License");  you may not use this file except in compliance with the License.  You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing, software  distributed under the License is distributed on an "AS IS" BASIS,  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  See the License for the specific language governing permissions and  limitations under the License.  -->  <android.support.v4.view.ViewPager xmlns:android=**"http://schemas.android.com/apk/res/android"**  android:id=**"@+id/pager"**  android:layout\_width=**"match\_parent"**  android:layout\_height=**"match\_parent"** /> |

login\_layout.xml

|  |
| --- |
| <RelativeLayout xmlns:android=**"http://schemas.android.com/apk/res/android"**  xmlns:tools=**"http://schemas.android.com/tools"**  android:layout\_width=**"match\_parent"**  android:layout\_height=**"match\_parent"** >  <EditText  android:id=**"@+id/username"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignParentLeft=**"true"**  android:layout\_alignParentRight=**"true"**  android:layout\_alignParentTop=**"true"**  android:ems=**"10"**  android:hint=**"Username"**  android:text=**"kambing"**>  <requestFocus />  </EditText>  <EditText  android:id=**"@+id/password"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignLeft=**"@+id/username"**  android:layout\_alignRight=**"@+id/username"**  android:layout\_below=**"@+id/username"**  android:ems=**"10"**  android:hint=**"Password"**  android:inputType=**"textPassword"**  android:text=**"kambing"**/>  <Button  android:id=**"@+id/sign"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignRight=**"@+id/username"**  android:layout\_below=**"@+id/password"**  android:layout\_marginTop=**"18dp"**  android:text=**"Sign In"** />  </RelativeLayout> |

task\_form\_layout.xml

|  |
| --- |
| <RelativeLayout  xmlns:android=**"http://schemas.android.com/apk/res/android"**  xmlns:tools=**"http://schemas.android.com/tools"**  android:layout\_width=**"fill\_parent"**  android:layout\_height=**"fill\_parent"**  android:padding=**"10dp"**  tools:context=**".TaskFormFragmentActivity"**  >  <EditText  android:id=**"@+id/editText1"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignParentLeft=**"true"**  android:layout\_alignParentRight=**"true"**  android:layout\_alignParentTop=**"true"**  android:ems=**"10"**  android:gravity=**"top|left"**  android:inputType=**"textMultiLine"**  android:lines=**"5"**  android:scrollHorizontally=**"false"**  android:background=**"@android:drawable/editbox\_background\_normal"**  >  <requestFocus />  </EditText>  <Button  android:id=**"@+id/button1"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignLeft=**"@+id/editText1"**  android:layout\_below=**"@+id/editText1"**  android:layout\_marginTop=**"10dp"**  android:text=**"Set Deadline"**  android:onClick=**"showDatePickerDialog"**/>  <TextView  android:id=**"@+id/textView1"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"wrap\_content"**  android:layout\_alignBottom=**"@+id/button1"**  android:layout\_alignRight=**"@+id/editText1"**  android:layout\_alignTop=**"@+id/button1"**  android:layout\_toRightOf=**"@+id/button1"**  android:textAppearance=**"?android:attr/textAppearanceMedium"**  android:text=**""**  android:gravity=**"center\_vertical"**/>  </RelativeLayout> |

task\_list.xml

|  |
| --- |
| <ListView xmlns:android=**"http://schemas.android.com/apk/res/android"**  android:id=**"@android:id/list"**  android:layout\_width=**"fill\_parent"**  android:layout\_height=**"fill\_parent"**  android:choiceMode=**"multipleChoice"**>  </ListView> |

task\_list\_row.xml

|  |
| --- |
| <LinearLayout xmlns:android=**"http://schemas.android.com/apk/res/android"**  android:layout\_width=**"fill\_parent"**  android:layout\_height=**"50dp"**  android:gravity=**"center\_vertical"**  >  <TextView  android:id=**"@+id/row\_day"**  android:layout\_width=**"60dp"**  android:layout\_height=**"wrap\_content"**  android:gravity=**"center\_vertical|center\_horizontal"**  android:textSize=**"25dp"**  android:textColor=**"@android:color/white"**  android:padding=**"10dp"**/>  <TextView  android:id=**"@+id/row\_title"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"match\_parent"**  android:layout\_weight=**"0.88"**  android:gravity=**"center\_vertical"**  android:paddingLeft=**"10dp"**  android:singleLine=**"true"**  android:textAppearance=**"?android:attr/textAppearanceMedium"** />  <CheckBox  android:id=**"@+id/row\_checkbox"**  android:layout\_width=**"wrap\_content"**  android:layout\_height=**"match\_parent"**  android:focusable=**"false"**/>  </LinearLayout> |