



# TASK MANAGER

Кире Петковски 185045

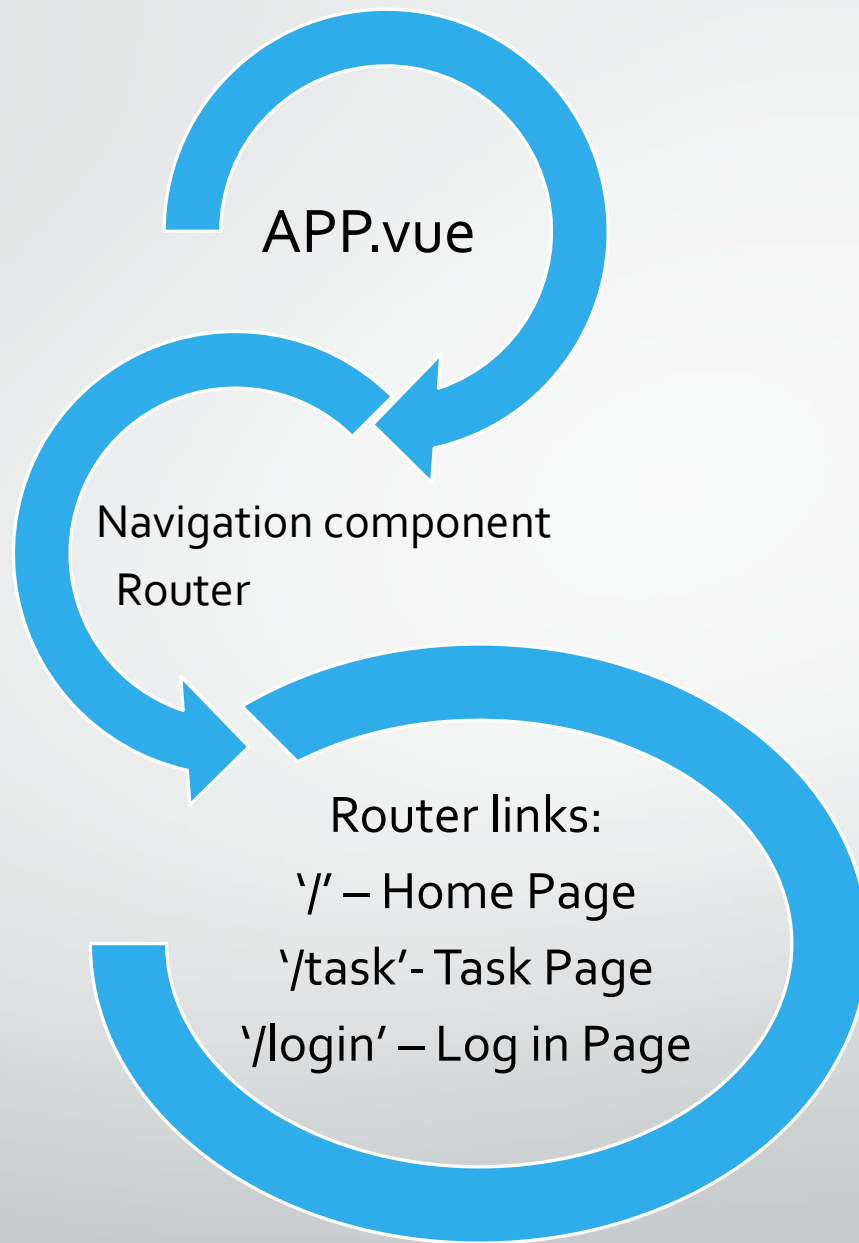
# Вовед

- Системот за менаџирање на задачи служи за додавање на задачи и нивно ажурирање низ текот на апликацијата.
- Погледот на апликацијата е поделен на администраторски и кориснички поглед.
  - Корисникот може ја промени состојбата на задачот и да додаде коментар на задачот.
  - Корисникот има опција да земе задача, но може и да му биде доделен од страна на администраторот.
  - Администраторот додава задачи во системот, но има и функции за бришење и ажурирање на задачите.
  - Администраторот може ја промени состојбата на задачот и да додаде коментар на задачот.
  - Администраторот има опција да земе задача, но може и сам да со додели задача.
  - Администраторот има поглед до сите задачи, додека корисникот има поглед само до задачите кои му се доделени до него.

# Router

- Корисниците кои не се најавени во системот неможат да пристапат до патеката 'localhost:8080/task'. За таа цел во дефинирањето на патеките, поставен е објект *meta* кој ја содржи променливата *requiresVisitor*.
  - *requiresVisitor* е *false* за патеката '/task', додека за останатите е *true*.
- При секоја нова промена на патеката, ќе провери дали корисникот е автентизиран.
  - Доколку не е најавен во системот ќе провери дали пристапува до патека која му е дозволена, инаку ќе го пренасочи кон патеката "localhost:8080/".

```
router.beforeEach( guard: (to : RouteLocationNormalized , from : RouteLocationNormalized , next : NavigationGuardNext ) => {  
  const isLoggedIn = window.localStorage.getItem( key: 'username' );  
  if (isLoggedIn) next()  
  else{  
    if(to.meta.requiresVisitor) next()  
    else next('/')  
  }  
})
```



'/' – Default path

' /task' – Дозволена само за автентифицирани корисници

# Navigation

TASK APP |

HOME

TASK

LOG OUT



TASK APP |



HOME

TASK

LOG OUT



```
<ul v-show="!mobile" class="navigation">
  <li>
    <router-link class="router-link" :to="{name: 'Home'}"> Home </router-link>
  </li>
  <li>
    <router-link class="router-link" :to="{name: 'Task'}"> Task </router-link>
  </li>
  <li v-show="!log">
    <router-link class="router-link" :to="{name: 'LogIn'}"> Log in </router-link>
  </li>
  <li v-show="log">
    <button id="logoutBorder" class="router-link" @click="Logout()"> Log out</button>
  </li>
</ul>
<div class="icon">
  <i @click="toggleMobileNav()" v-show="mobile" class="fa fa-bars" :class="{ 'icon-active': mobileNav }"></i>
</div>
<transition name="mobile-nav">
  <ul v-show="mobileNav" class="dropdown-nav">
    <li>
      <router-link class="router-link" :to="{name: 'Home'}"> Home </router-link>
    </li>
    <li>
      <router-link class="router-link" :to="{name: 'Task'}"> Task </router-link>
    </li>
    <li v-show="!log">
      <router-link class="router-link" :to="{name: 'LogIn'}"> Log in </router-link>
    </li>
    <li v-show="log">
      <button id="logoutBorderMobile" class="router-link" @click="Logout()"> Log out</button>
    </li>
  </ul>
</transition>
```

# Firebase - Configuration

```
import firebase from 'firebase';
import 'firebase/firestore'

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "",
  authDomain: "",
  projectId: "",
  storageBucket: "",
  messagingSenderId: "",
  appId: "",
  measurementId: ""
};

// Initialize Firebase
firebase.initializeApp(firebaseConfig);
const db = firebase.firestore();

export default db;
```







# Firebase - Authentication

- Firebase има посебна алатка за автентикација. Содржи табела од корисници.
  - Табелата ја пополнуваме со најава на email и password.
  - Секој корисник има уникатен ID кој се креира од страна на Firebase.
- Колекцијата Users ни служи за одредување на корисничкото име и улогата на корисникот.
- Документите од колекцијата Users имаат ID кое е исто со UID од табелата на автентикација.
  - Со најава на корисникот се презима UID и потоа се проверува корисничко име и улогата

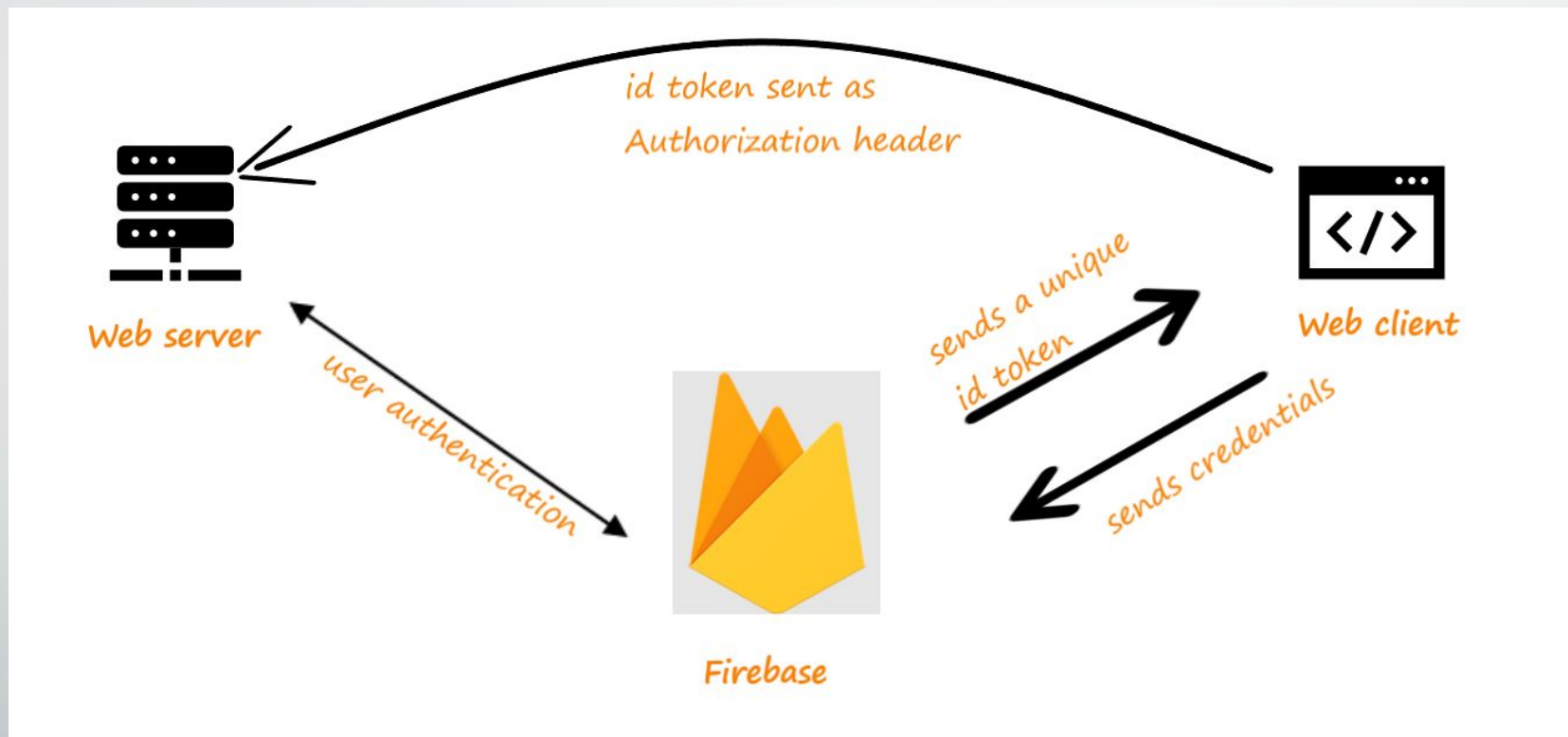


# Firebase - Authentication

Identifier	Providers	Created ↓	Signed In	User UID
user2@example.com		Feb 3, 2023	Feb 6, 2023	dNYjaf9pkmhoKUK6NqiQYycurDh1
user1@example.com		Feb 3, 2023	Feb 13, 2023	88mdC2io4LgqvT4ZZrux3fwls0o2
admin@example.com		Feb 3, 2023	Feb 13, 2023	EPOIrVyBJaY674FFSAZAbyN4bNo1

 first-vueapp-test	 Users  	 88mdC2io4LgqvT4ZZrux3fwls0o2 
<a href="#">+ Start collection</a>	<a href="#">+ Add document</a>	<a href="#">+ Start collection</a>
Tasks	88mdC2io4LgqvT4ZZrux3fwls0o2 >	<a href="#">+ Add field</a>
Users >	EPOIrVyBJaY674FFSAZAbyN4bNo1	Role: "user"
	dNYjaf9pkmhoKUK6NqiQYycurDh1	Username: "User1"

# Firestore - Authentication



# Log In

```
login() {
  this.Request = true;
  this.errorMessage = "";

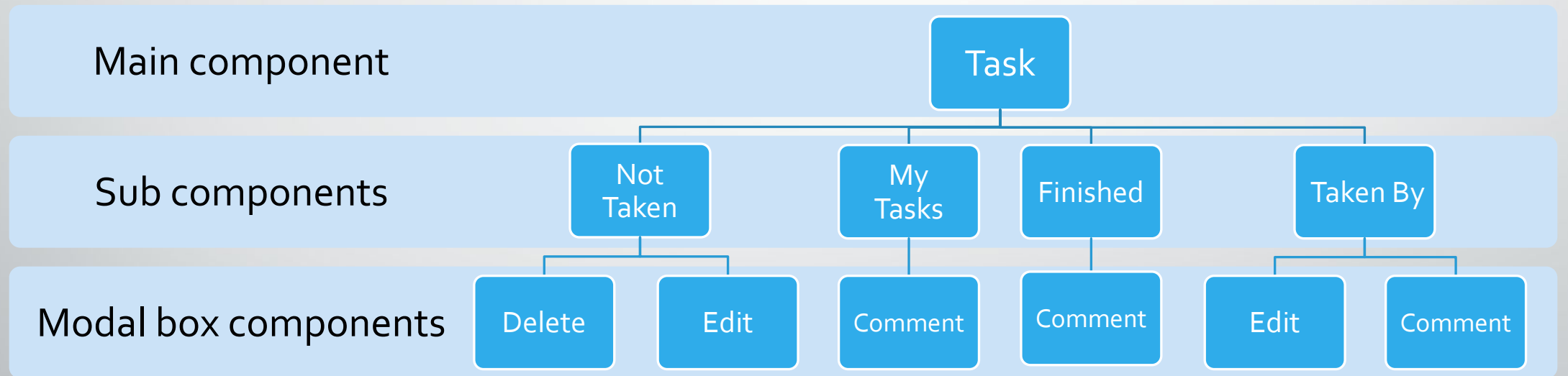
  firebase.auth().signInWithEmailAndPassword(this.email, this.password).then(
    (cred) => {
      db.collection('collectionPath: "Users").doc(cred.user.uid).get().then((doc) => {

        this.user.username = doc.data().Username;
        this.user.role = doc.data().Role;

        window.localStorage.setItem('username', this.user.username);
        window.localStorage.setItem('role', this.user.role);

        this.Request = false;
        this.$router.go( delta: 0);
      })
    },
    (error) => {
      this.errorMessage = error.message;
      this.Request = false;
    }
  ).then(
    this.$router.replace( to: "/" )
  )
},
```

# Task page



# Task component

- *Task* компонентата ги презима податоците од *Firebase* и ги предава на под компонентите.
  - Чува податоци од документите од *Firebase: Tasks* и *Users*
- Податоците во *task* компонентата се ажурираат со секоја промена на под компонентите
- Во `create()` е поставен делот со кој компонентата ги презима податоците од `firebase`.
- `Add()` е метода која од формата го презима објаснувањето за таскот и креира објект кој е потребен за да се прати до `firebase` за да се сочува таскот.

# Task component - Firebase

first-vueapp-test	Tasks	u0MDn3CVxU6TadLA0egw
+ Start collection	+ Add document	+ Start collection
Tasks >	9QGXMqZpXUKIfWsW82y7	+ Add field
Users	r3fax9gFPmK4cJACPyQZ	▼ Comment
	u0MDn3CVxU6TadLA0egw >	0 "First comment"
		1 "Second comment"
		2 "Third comment"
		Description: "First task"
		ID: 1
		Status: "In Progress"
		User: "User1"

first-vueapp-test	Users	EP0IrVyBJaY674FFSAZAbyN4bNo1
+ Start collection	+ Add document	+ Start collection
Tasks	88mdC2io4LgqvT4ZZrux3fwls0o2	+ Add field
Users >	EP0IrVyBJaY674FFSAZAbyN4bNo1 >	Role: "admin"
	dNYjaf9pkmhoKUK6NqiQYycurDh1	Username: "Admin"

# Task component

```
methods: {
  addTask() {
    // console.log("Successfully added new task: " + this.newTaskDescription);
    if (this.newTaskDescription.length > 0) {
      if (this.givenUser !== "") {
        this.status = "In Progress";
      } else {
        this.status = "Not Taken";
      }
      const task = {
        'Comment': [],
        'Description': this.newTaskDescription,
        'ID': this.NextIDTask + 1,
        'User': this.givenUser,
        'Status': this.status,
      }
      db.collection( collectionPath: "Tasks").add(task)
        .then(
          this.tasks = []
        );
    }
    this.newTaskDescription = "";
    this.givenUser = "";
  },
}
```

```
created() {
  db.collection( collectionPath: "Users").get().then((querySnapshot) => {
    querySnapshot.forEach( callback: (doc) => {
      var getUser = {
        'username': doc.data().Username,
      }
      this.Users.push(getUser);
    });
  });

  db.collection( collectionPath: "Tasks").onSnapshot( onNext: (querySnapshot) => {
    querySnapshot.forEach( callback: (doc) => {
      if (!this.tasks.find(({id}) => id === doc.id)) {
        const task = {
          'id': doc.id,
          'task_id': doc.data().ID,
          'description': doc.data().Description,
          'status': doc.data().Status,
          'user': doc.data().User,
          'comments': doc.data().Comment,
        }
        console.log("CREATE for: " + this.user.username + " " + this.user.role);
        this.tasks.push(task);
      }
      if (this.NextIDTask <= doc.data().ID) {
        this.NextIDTask = doc.data().ID;
      }
    });
  });
},
```



# Task component

```
<nav>
  <div class="nav nav-tabs" id="nav-tab" role="tablist">
    <button class="nav-link" type="button" role="tab"
      @click="ChangeTab( TabName: 'New tasks')">New tasks
    </button>
    <button class="nav-link" type="button" role="tab"
      @click="ChangeTab( TabName: 'My tasks')">My tasks
    </button>
    <button class="nav-link" type="button" role="tab"
      @click="ChangeTab( TabName: 'Finished tasks')">Finished tasks
    </button>
    <button class="nav-link" type="button" role="tab" v-show="user.role === 'admin'"
      @click="ChangeTab( TabName: 'Task in Progress')">Task in Progress
    </button>
  </div>
</nav>
```

```
<div class="tab-content" id="nav-tabContent">
  <div v-if="activeTab === 'New tasks'" role="tabpanel">
    <NotTaken :tasks="tasks.sort( compareFn: (a, b) => b.task_id - a.task_id)"
      :role="user.role" :user="user.username"
      @deleteTasks=" this.tasks.splice(index, deleteCount: 1);"
      @changeStatus=" this.tasks[$event].status = 'In Progress'"
      @changeUser="this.tasks[$event.tasksIndex].user = $event.user"
      @editTask="this.tasks[$event.tasksIndex].description = $event.description"
    </div>
  <div v-if="activeTab === 'My tasks'" role="tabpanel">
    <MyTasks :tasks="tasks.sort( compareFn: (a, b) => b.task_id - a.task_id)"
      :user="user.username"
      @closeTask="this.tasks[$event.tasksIndex].status = $event.status"
    </div>
  <div v-if="activeTab === 'Finished tasks'" role="tabpanel">
    <Finished :tasks="tasks.sort( compareFn: (a, b) => b.task_id - a.task_id)"
      :role="user.role" :user="user.username"
    </div>
  <div v-if="activeTab === 'Task in Progress'" role="tabpanel">
    <TakenBy :tasks="tasks.sort( compareFn: (a, b) => b.task_id - a.task_id)"
      @editTask="this.tasks[$event.tasksIndex].description = $event.description"
    </div>
</div>
```



# Task page – Not Taken component

- Во табеларниот приказ ќе се појави ново креиран таск ако не се додели на некој корисник.
  - Статус: Not Taken
- Погледот на не земени таскови можат да пристапат сите корисници.
  - Администраторот може да го земе таскот, да го избрише или да го ажурира описот на таскот.
  - Корисникот има опција на презимање на таск.

# Task page

## Администраторски поглед

<a href="#">New tasks</a> <a href="#">My tasks</a> <a href="#">Finished tasks</a> <a href="#">Task in Progress</a>					
ID	Task	Status	Take task	Delete	Edit
5	Fifth for testing task app	Not Taken	+		

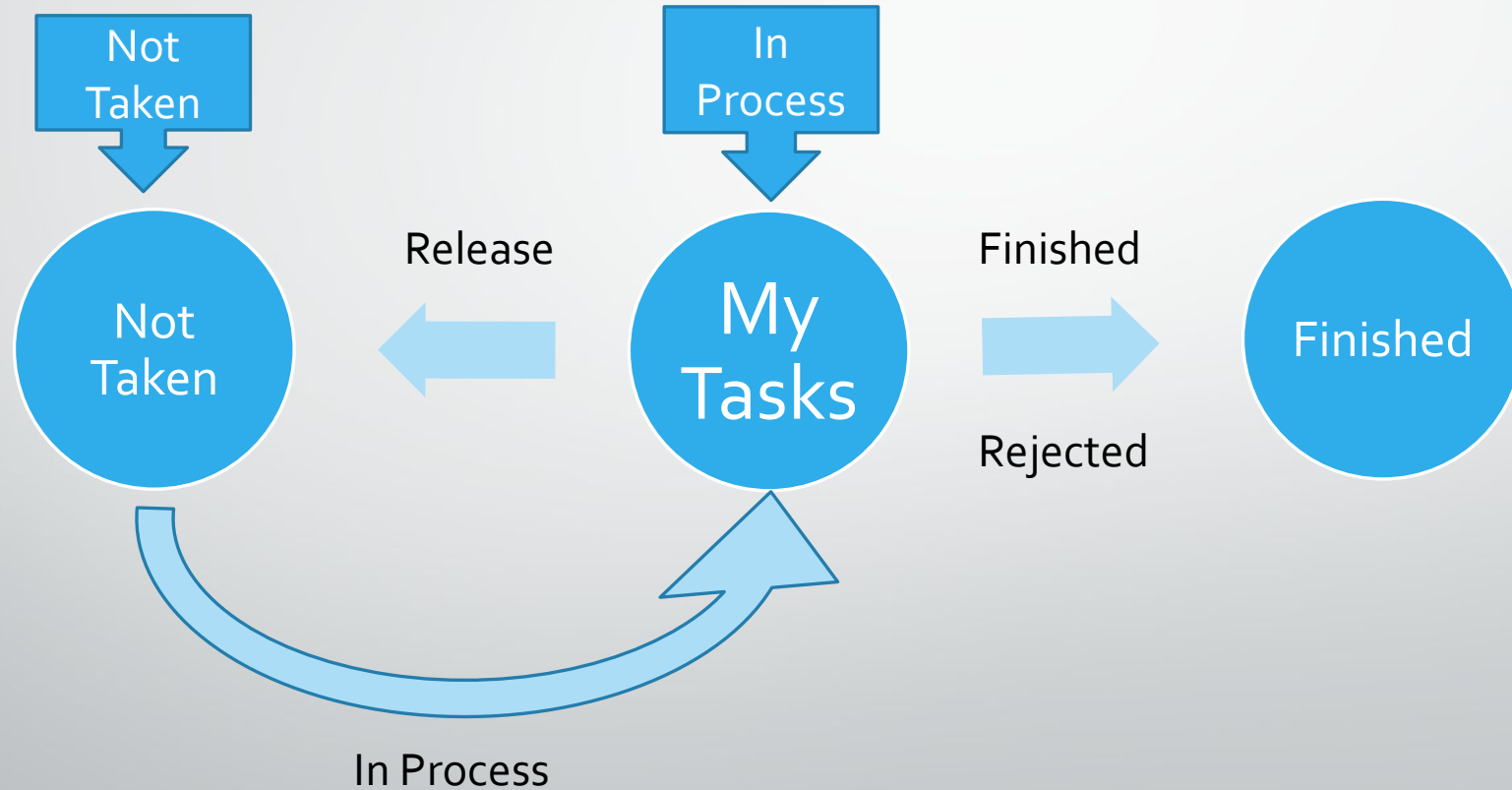
## Кориснички поглед

<a href="#">New tasks</a> <a href="#">My tasks</a> <a href="#">Finished tasks</a>			
ID	Task	Status	Take task
5	Fifth for testing task app	Not Taken	+

# Task page – My tasks component





- Во табот My Tasks корисниците ги гледаат само тасковите кои им доделени.
- Во овој дел корисникот е должен да го промени статусот, да додаде коментар и да го затвори таскот.
  - Статусот може да се промени со кликање врз текстот.
- Статус:
  - In Process: Таскот не е завршен. Неможе да се затвори таскот без предходно да се промени.
  - Release: Таскот е ослободен за да го преземе некој друг. Таскот се враќа во табот Not Taken.
  - Rejected: Таскот е затворен, но не може да се заврши поради одредена пречка.
  - Finished: Таскот е затвотрен и успешно е средена работата зададена во описот.

# Task page – My tasks component



# Task page

Администраторски и кориснички поглед

<a href="#">New tasks</a> <a href="#">My tasks</a> <a href="#">Finished tasks</a> <a href="#">Task in Progress</a>				
ID	Task	Status	Comment	Close
4	Nov task	In Progress 		
2	Task for Admin	In Progress		

Грешка при затварање на task – статус In progress треба да се проомени

# Task page

Администраторски поглед:


New tasks	My tasks	Finished tasks	Task in Progress	
ID	Task	Status	User	Comment
7	Task for User1	Finished	User1	
6	Task for User2	Finished	User2	
2	Task for Admin	Rejected	Admin	

Кориснички поглед:

New tasks

My tasks

Finished tasks

ID	Task	Status	Comment
7	Task for User1	Finished	

# Task page – In Progress component

- Табот In Progress може да пристапи само администраторот.
- На овој таб се прикажуваат сите задачи кои се преземени од некој корисник и сеуште не се затворени.
- Администраторот може да додаде коментар на задачот и може да го ажурира описот на задачот.

<a href="#">New tasks</a> <a href="#">My tasks</a> <a href="#">Finished tasks</a> <a href="#">Task in Progress</a>					
ID	Task	Status	User	Comment	Edit
4	Nov task	In Progress	Admin		
3	New task	In Progress	User1		
1	First task	In Progress	User1		