Deminder

An app that reminds you of your deadlines

Thomas Malina – Tillmann Nünninghof – Natalie Busam – Lea Wegner

Table of contents

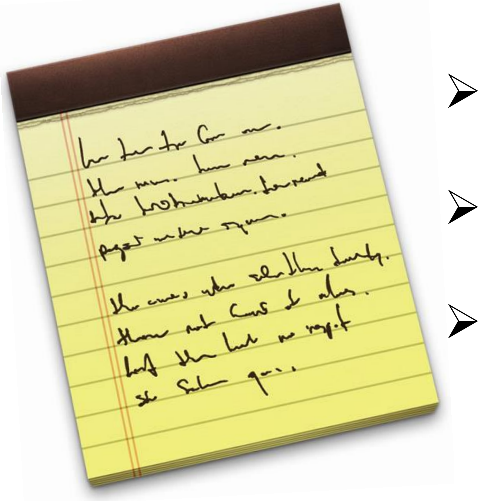
- Understand business need
- Technical Ability
- Quality

Understand business need

Vision

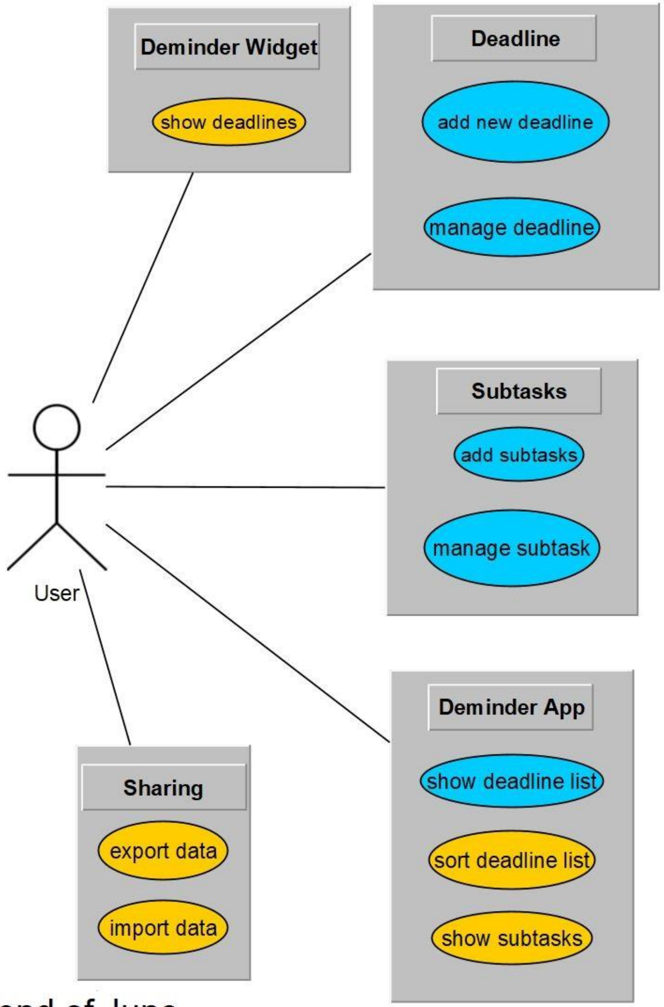
- Many deadlines to keep track off
- Exams, dates, papers you need to finish and e-mails you have to write
- Focus on important things

- Show an overview of your deadlines
- The ability to create, manage and delete deadlines and subtasks
- Subtask with check option



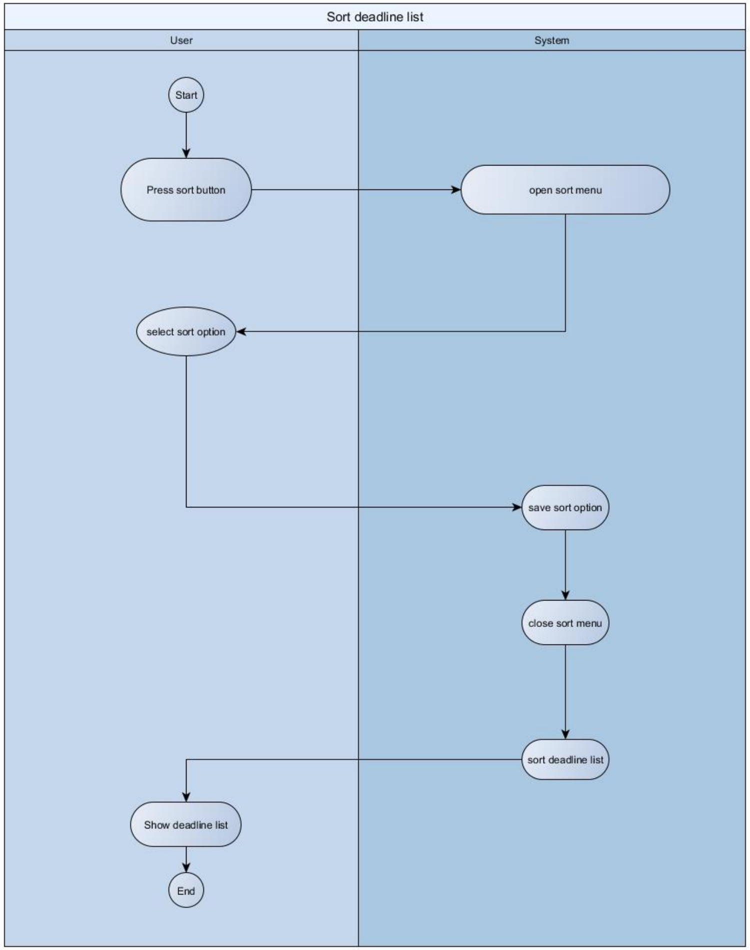
- Possibility to set deadlines as recurring
- Don‘t forget any important upcoming dates
- Organize and track your progress

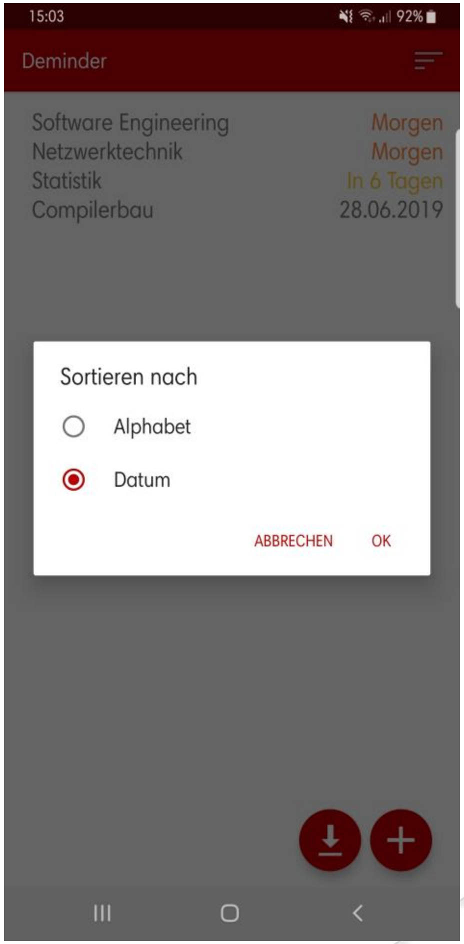
Use cases



🟨 will be implemented till end of June

Use case – Sort deadline list





Software Engineering	Morgen
Netzwerktechnik	Morgen
Statistik	In 6 Tagen
Compilerbau	28.06.2019

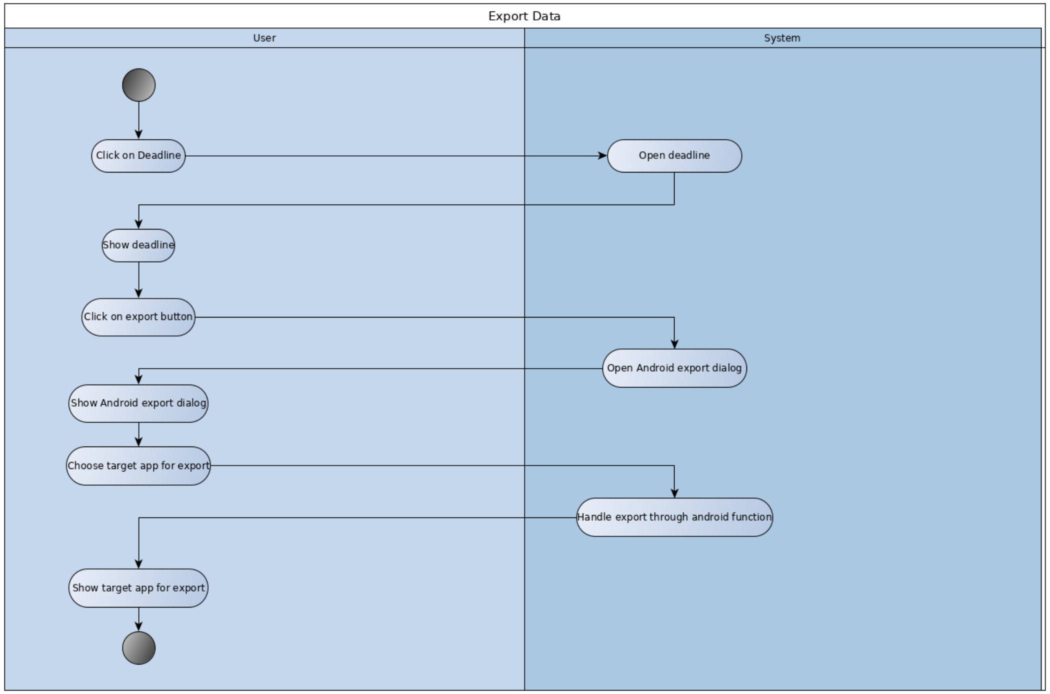
Sortieren nach

○ Alphabet

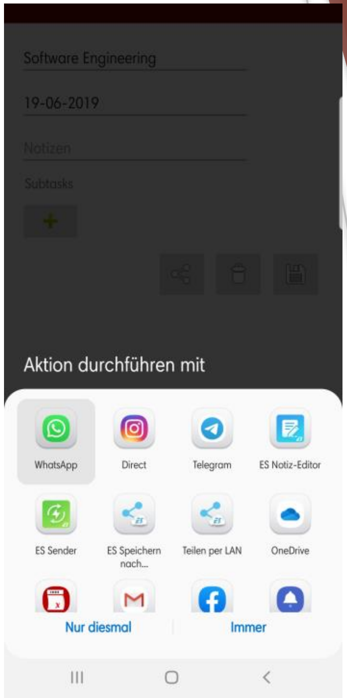
● Datum

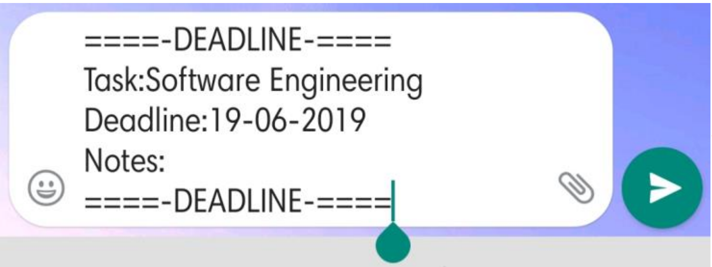
ABBRECHEN OK

Use case – Export data



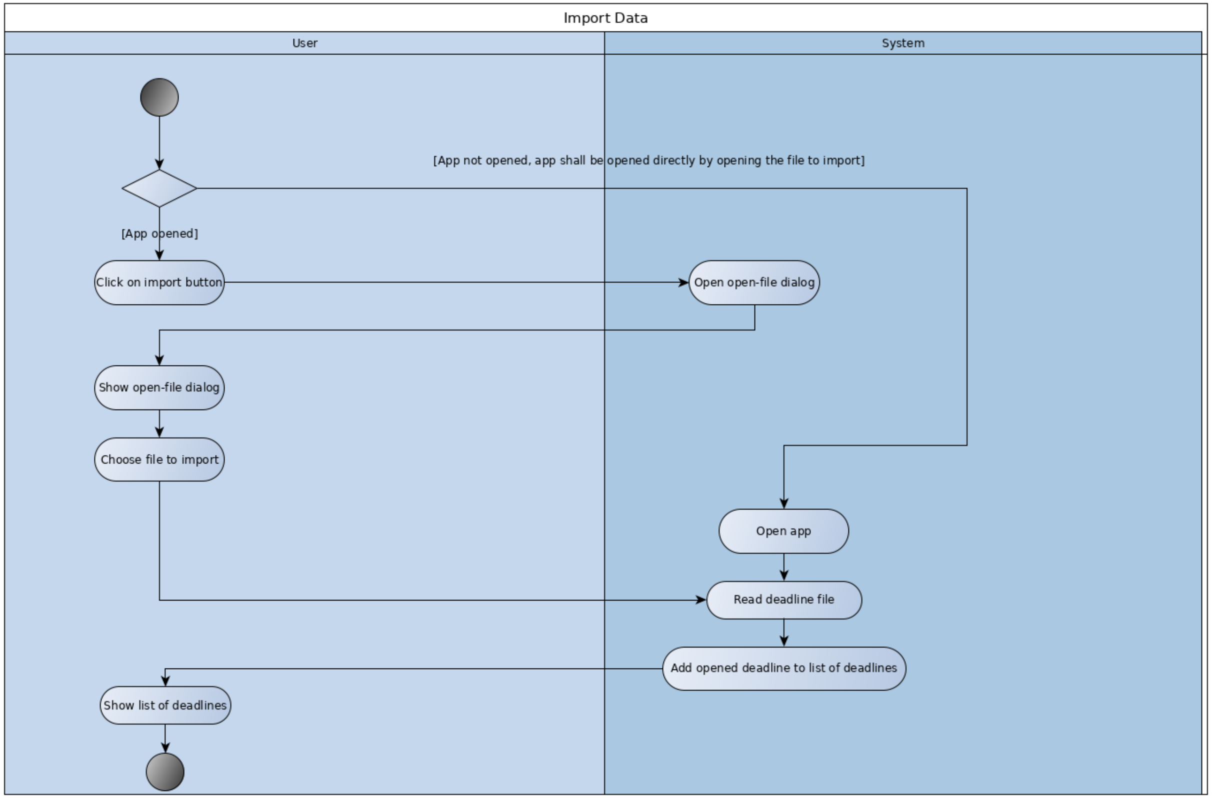
Export Data
User
System
Click on Deadline
Open deadline
Show deadline
Click on export button
Open Android export dialog
Show Android export dialog
Choose target app for export
Handle export through android function
Show target app for export





====-DEADLINE-====
Task:Software Engineering
Deadline:19-06-2019
Notes:
====-DEADLINE-====

Use case – Import data



Import Data

User

System

[App not opened, app shall be opened directly by opening the file to import]

[App opened]

Click on import button

Open open-file dialog

Show open-file dialog

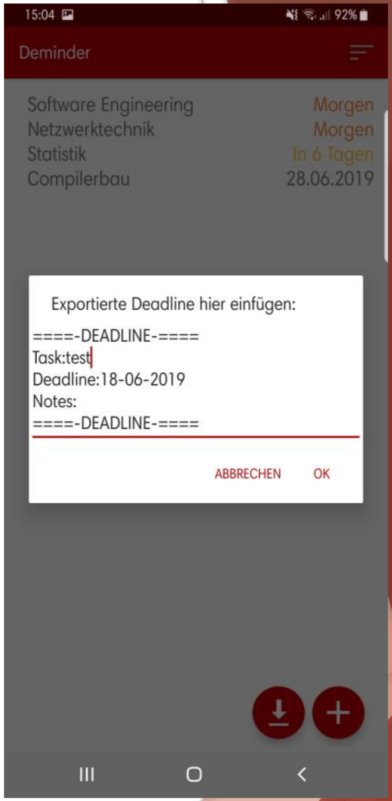
Choose file to import

Open app

Read deadline file

Add opened deadline to list of deadlines

Show list of deadlines



Software Engineering Morgen

Netzwerktechnik Morgen

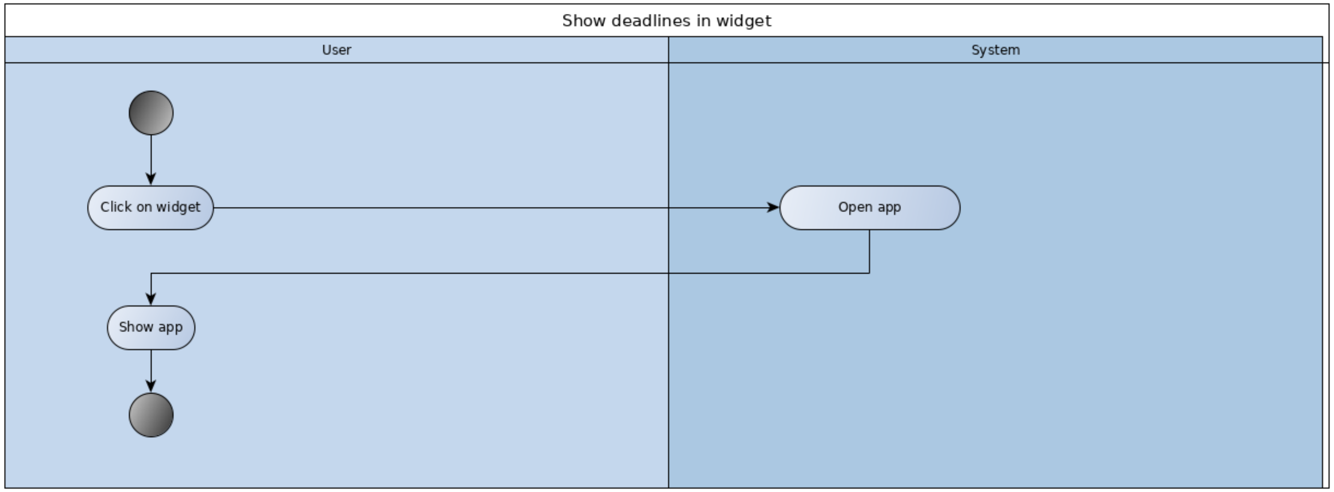
Statistik In 6 Tagen

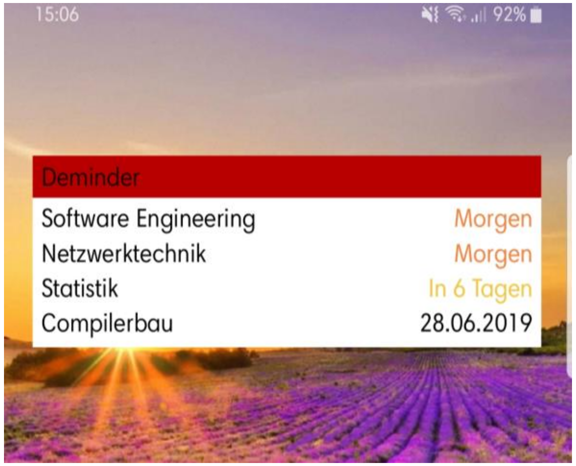
Compilerbau 28.06.2019

Exportierte Deadline hier einfügen:

```
====-DEADLINE-====
Task:test
Deadline:18-06-2019
Notes:
====-DEADLINE-====
```

Use case – Widget





Deminder

Software Engineering	Morgen
Netzwerktechnik	Morgen
Statistik	In 6 Tagen
Compilerbau	28.06.2019

Software Requirements Specification

Scope:

- remind the user of upcoming deadlines such as exams
- overview to show all your deadlines

Functionality:

- Create, edit, show, delete deadline
- Add, show and delete subtasks, mark subtasks as finished

Non-functionals

Reliability

- Data is never lost or corrupted, acurate time, app always active

Performance

- Fast loading, asynchronus saving, low response time

Supportability

- Conventions, clear programm structure supporting modularity, clean code and commits

Design constraints

- Java, Android Studio, Git, YouTrack, Wordpress

Project Methodology

Tool

- YouTrack

PM method

- Scrum

What is Scrum?

- breaks large processes down into small pieces in order to streamline efficiency
- Time-tracking

Advantages

- Very agile

Iterative Process

- Small Scrum tasks to complete a objective set for our sprint
- Each task is complete on its own and contributes towards the sprint goal

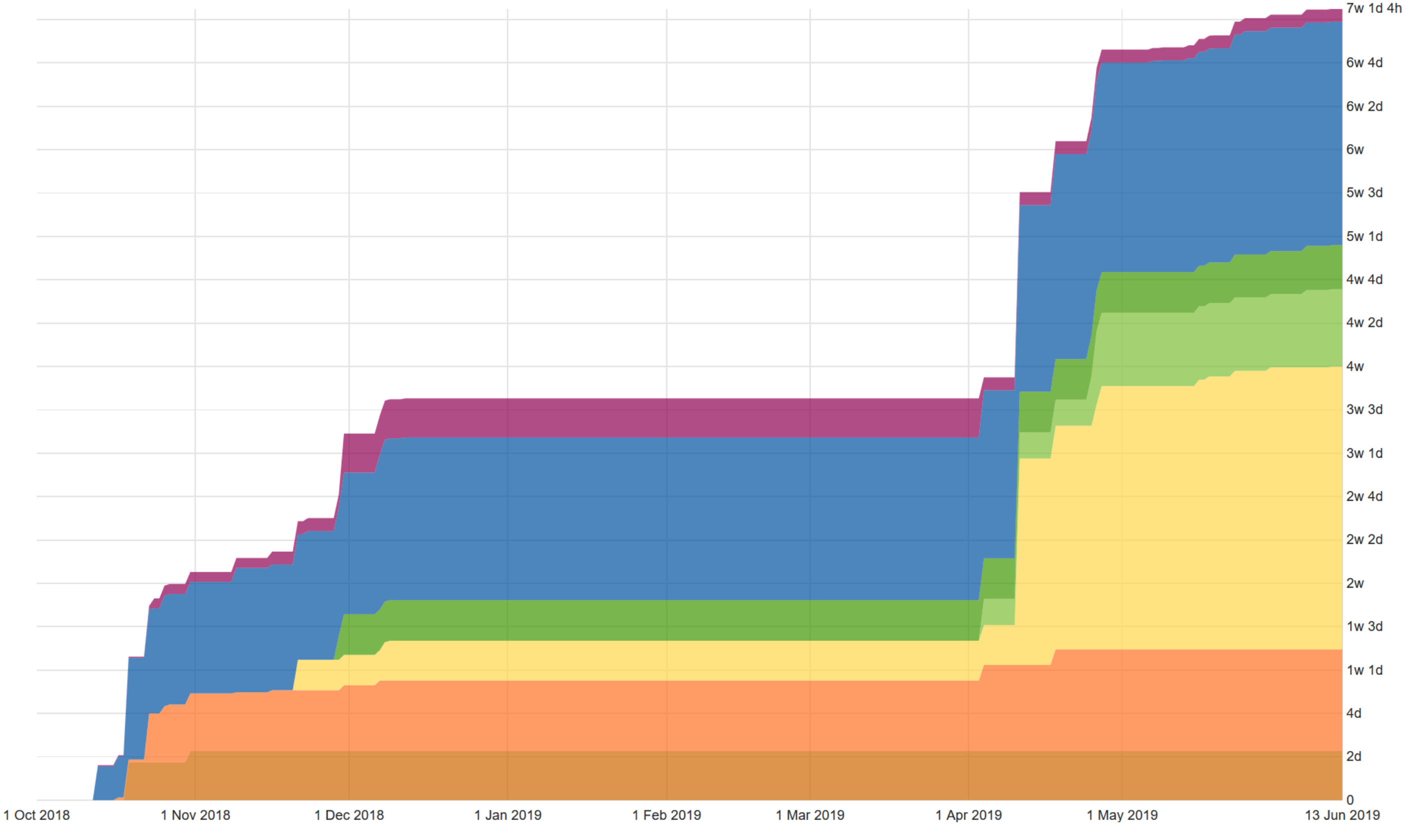
Project Management

- access our task list in our IDE, Android Studio
- track the time one of our team members is working on a specific task
- Contributing to our git repository via Android Studio directly

RUP and long-term planning

- Usage of RUP for clear roles of each member and structure
- Burndown to keep track of our progress

Cumulative Chart



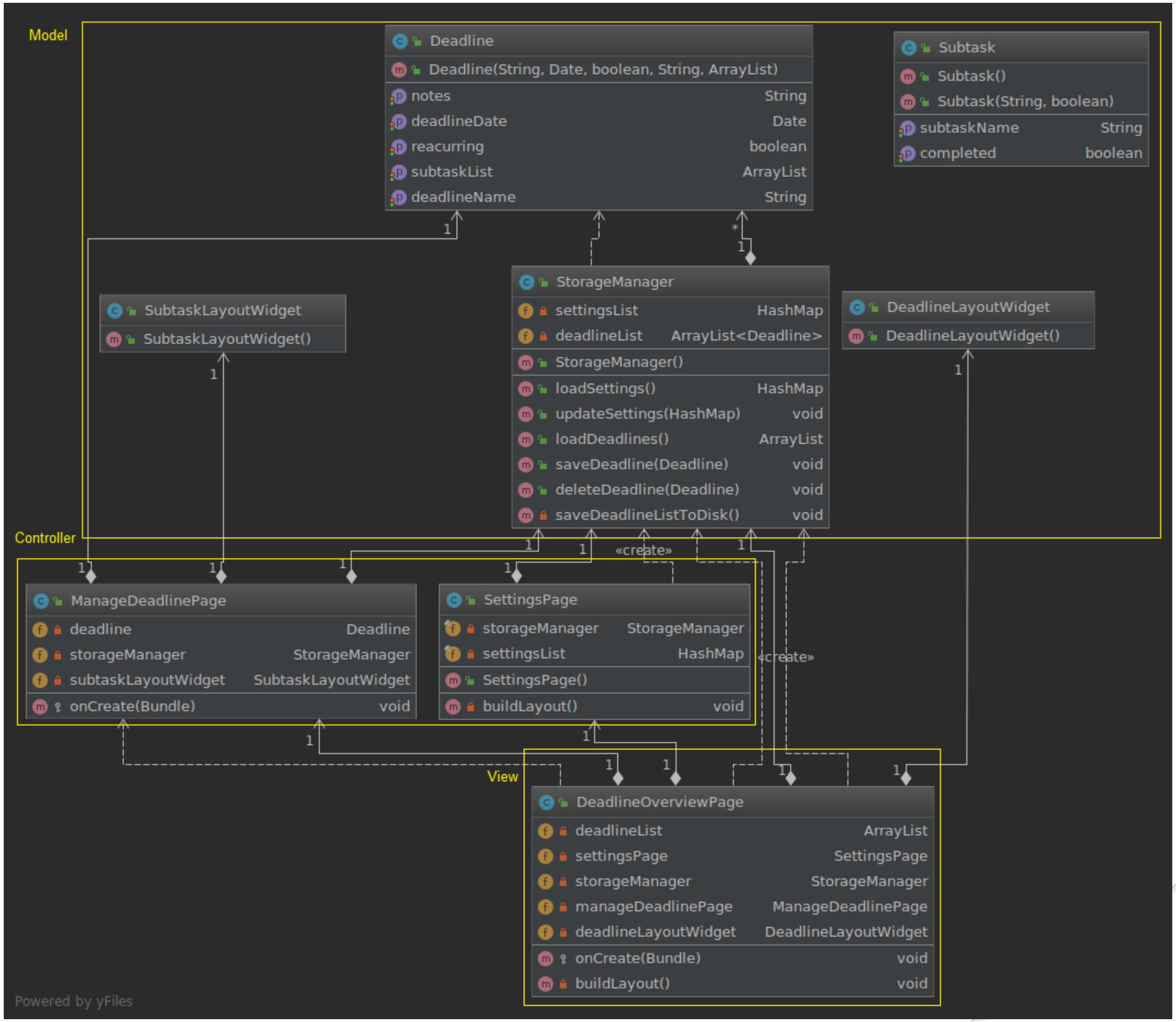
Cost Estimation

Use Case	FP's	Estimated Time	Time spent
Import Data	11,76	50	1d 6h
Export Data	11,76	50	6h
Sort deadline list	10,9	45	1d 3h
Show subtasks	5,88	20	1d 8h

Technical ability

LIVE DEMO

Class diagram



Deadline

Deadline(String, Date, boolean, String, ArrayList)	
notes	String
deadlineDate	Date
reacurring	boolean
subtaskList	ArrayList
deadlineName	String

Subtask

Subtask()	
Subtask(String, boolean)	
subtaskName	String
completed	boolean

StorageManager

settingsList	HashMap
deadlineList	ArrayList<Deadline>
StorageManager()	
loadSettings()	HashMap
updateSettings(HashMap)	void
loadDeadlines()	ArrayList
saveDeadline(Deadline)	void
deleteDeadline(Deadline)	void
saveDeadlineListToDisk()	void

SubtaskLayoutWidget

SubtaskLayoutWidget()	

DeadlineLayoutWidget

DeadlineLayoutWidget()	

ManageDeadlinePage

deadline	Deadline
storageManager	StorageManager
subtaskLayoutWidget	SubtaskLayoutWidget
onCreate(Bundle)	void

SettingsPage

storageManager	StorageManager
settingsList	HashMap
SettingsPage()	
buildLayout()	void

DeadlineOverviewPage

deadlineList	ArrayList
settingsPage	SettingsPage
storageManager	StorageManager
manageDeadlinePage	ManageDeadlinePage
deadlineLayoutWidget	DeadlineLayoutWidget
onCreate(Bundle)	void
buildLayout()	void

Quality

Architecture

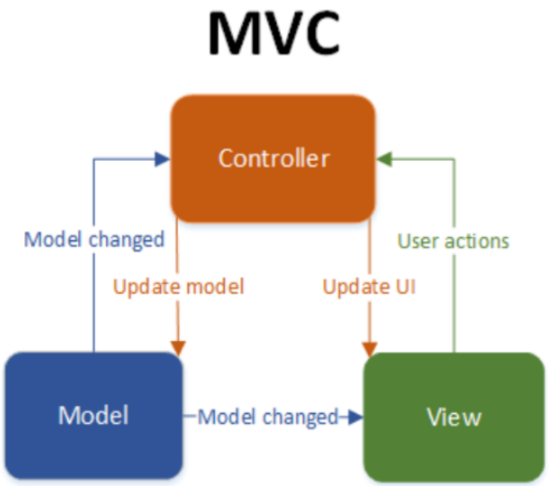
- No server, only a client side which uses MVC
- Written in Java for android

Model View Controller

- Goal: seperate view from logic
- Controller takes care of handling actions by the user
- Models contain the data that is displayed in the views

Data Storage

- Deadlines stored on the phone drive



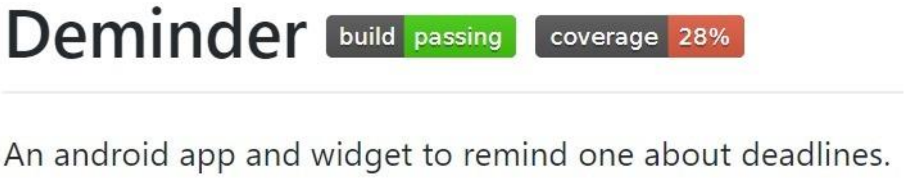
Automation

- Automatic time tracking
- Automatic tests
- Automatic deployment

Continuous Integration

- Travis CI as automated build service
- Every commit to the master branch triggers a build
- Travis CI runs all tests and statistics
- If build fails, commiter gets

Badges:

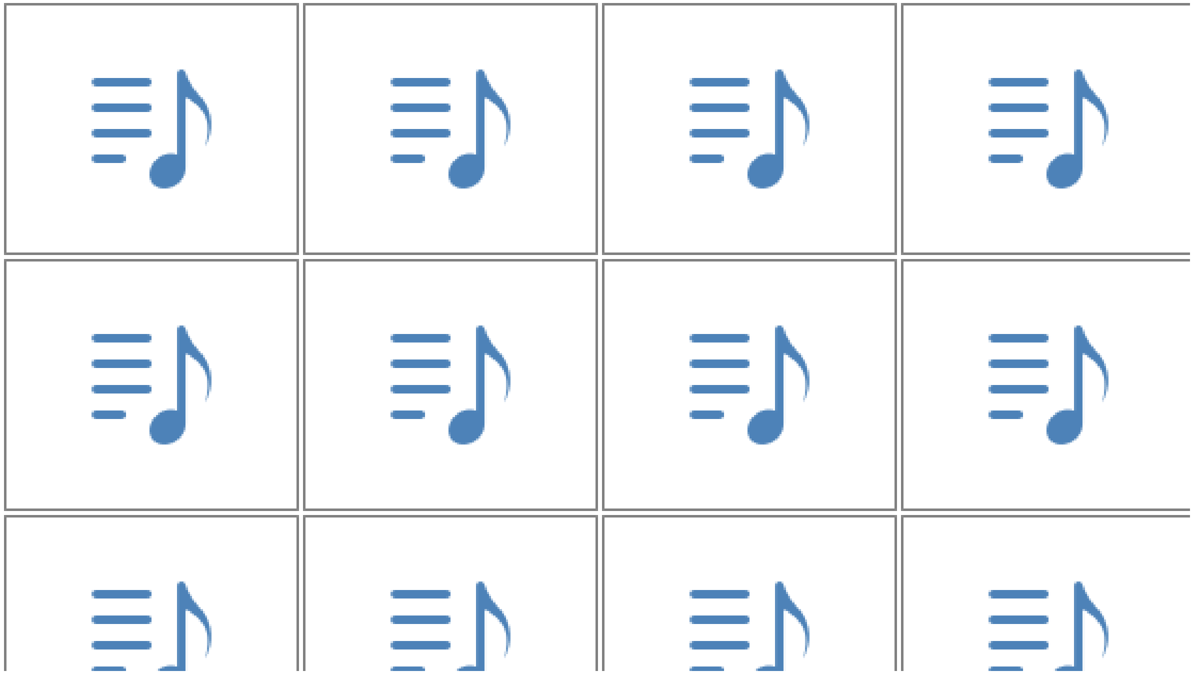


Deminder

An android app and widget to remind one about deadlines.

Testing

- Feature test with Espresso Framework



Testing – User tests

- **Functional Testing**: Checking all app content for correct functioning.
- **Usability-Test**: Testing the app‘s user-friendliness (especially layout and handling).
- **Compatibility-Test:** Check whether the app is compatible with different hardware and software conditions.
- **Performance-Test**: Review of the system resources required by the app during intensive use.
- **Functions-Test**: Check if all relevant functions of the app work.
- (Personal Feedback)

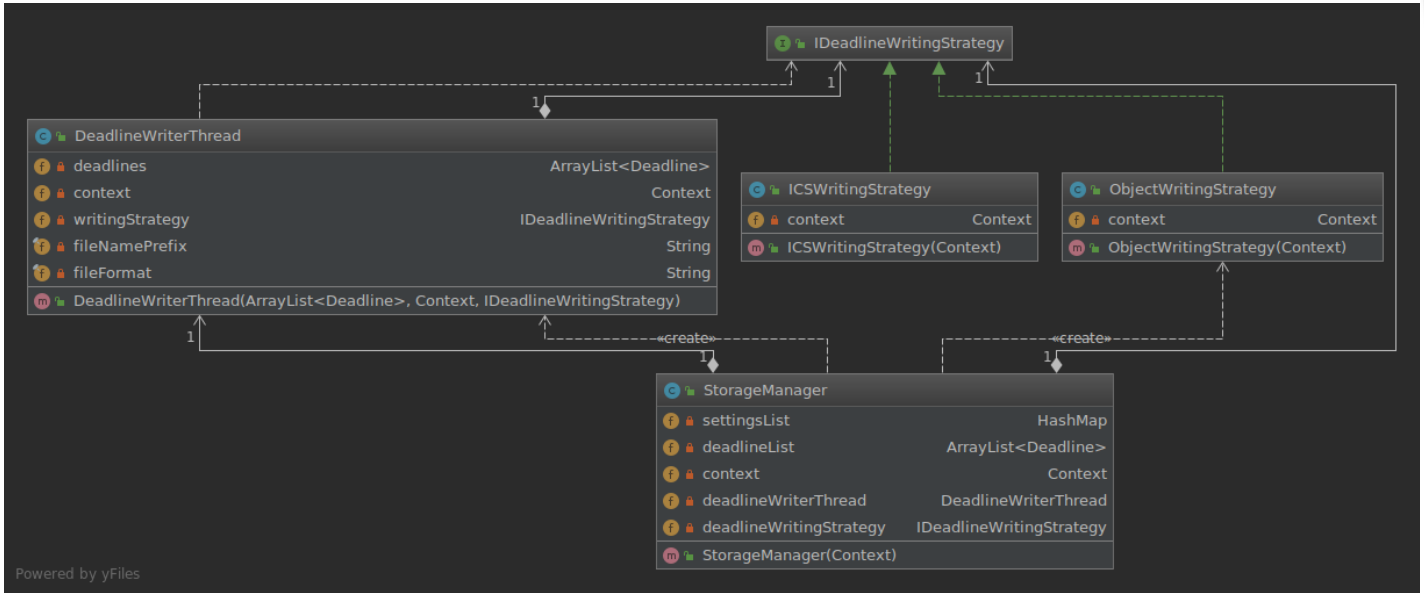
Risk Management

Name	Occurence	Impact	Factor
The speciality of the app is not clear	80%	8	6,4
Inadequate estimation of required resources	50%	7	3,5
High level of technical complexity	60%	5	3
Ineffective communications	50%	6	3

Risk Management

Name	Occurence	Impact	Factor
Unknown technologies are difficult to handle	90%	3	2,7
User does not understand the usage of the app	20%	8	1,6
Loss of team members	10%	10	1
Changing requirements	10%	8	0,8

Patterns



IDeadlineWritingStrategy

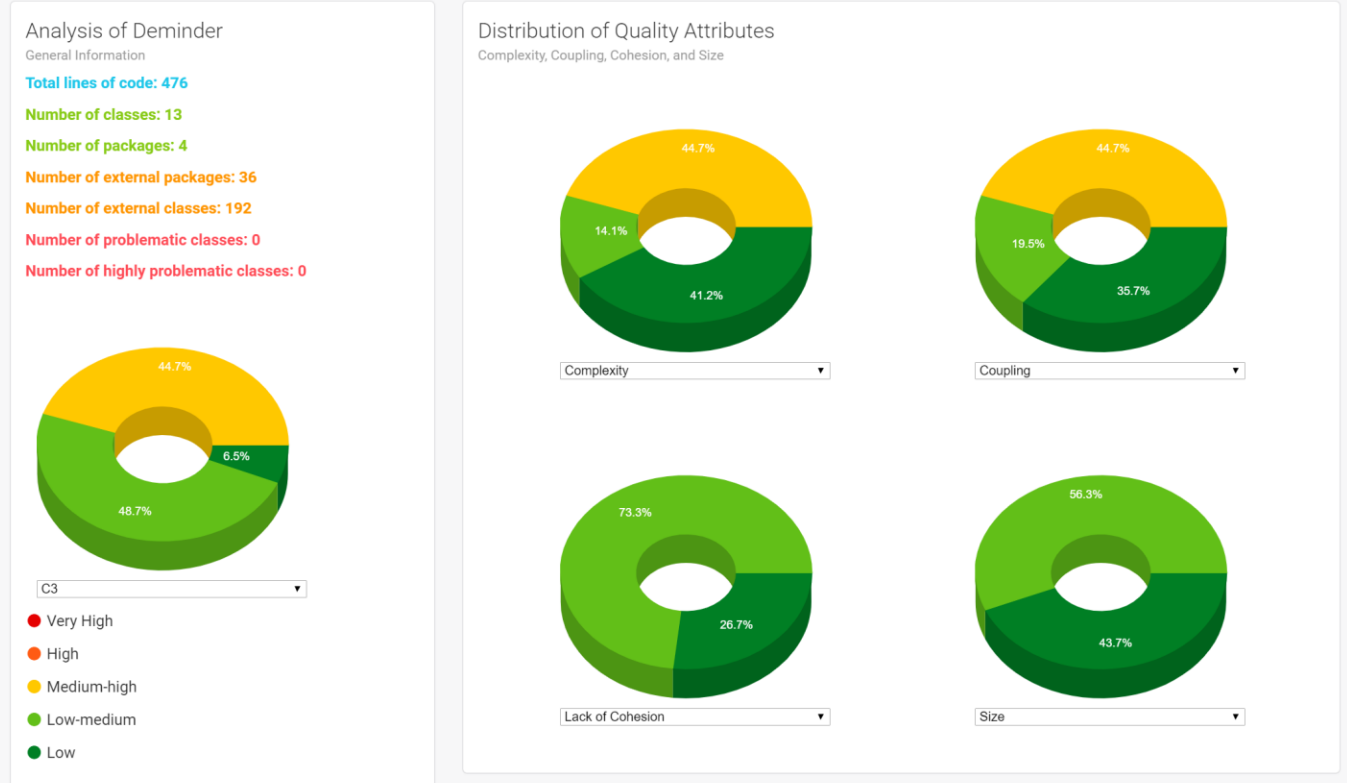
DeadlineWriterThread
deadlines ArrayList<Deadline>
context Context
writingStrategy IDeadlineWritingStrategy
fileNamePrefix String
fileFormat String
DeadlineWriterThread(ArrayList<Deadline>, Context, IDeadlineWritingStrategy)

ICSWritingStrategy
context Context
ICSWritingStrategy(Context)

ObjectWritingStrategy
context Context
ObjectWritingStrategy(Context)

StorageManager
settingsList HashMap
deadlineList ArrayList<Deadline>
context Context
deadlineWriterThread DeadlineWriterThread
deadlineWritingStrategy IDeadlineWritingStrategy
StorageManager(Context)

Metrics



Analysis of Deminder

General Information

Total lines of code: 476

Number of classes: 13

Number of packages: 4

Number of external packages: 36

Number of external classes: 192

Number of problematic classes: 0

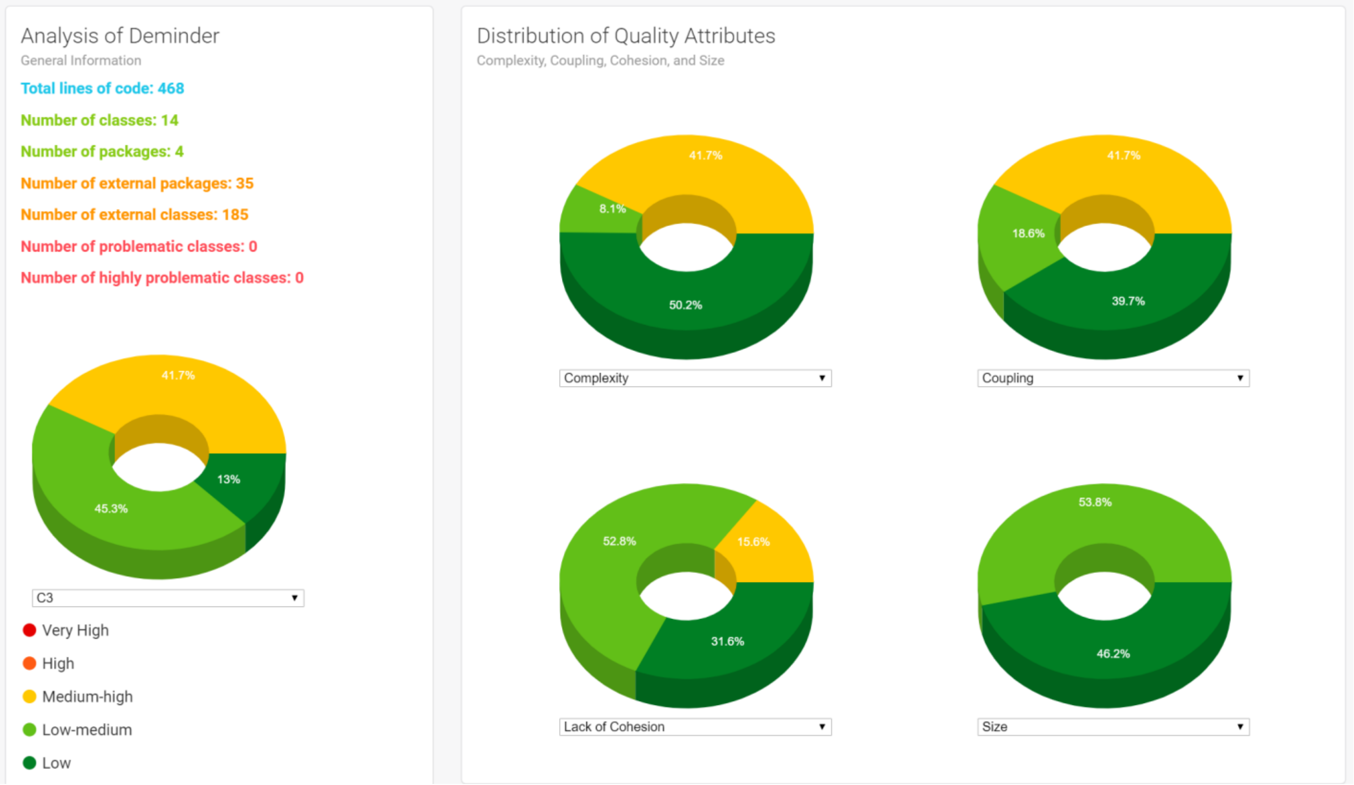
Number of highly problematic classes: 0

Distribution of Quality Attributes

Complexity, Coupling, Cohesion, and Size

- Complexity
- Coupling

Metrics



Analysis of Deminder

General Information

Total lines of code: 468

Number of classes: 14

Number of packages: 4

Number of external packages: 35

Number of external classes: 185

Number of problematic classes: 0

Number of highly problematic classes: 0

C3 ▾

- ● Very High
- ● High
- ● Medium-high
- ● Low-medium
- ● Low

Distribution of Quality Attributes

Complexity, Coupling, Cohesion, and Size

Complexity ▾

Coupling ▾

Lack of Cohesion ▾

Size ▾

References

- **Git-Repository:** https://github.com/Kalkihe/Deminder
- **Blog:** https://deadlinereminder.wordpress.com/
- **Download-Link:** https://deadlinereminder.wordpress.com/2019/06/11/installation-and-user-tests/

Thank you for your attention!