**Growth: A worklist program**

**Course Project of Java EE**

**1652694 Chen Lu**

**1652704 Wang Hanqiu**

**1652720 Li Yuhan**

**1652780 Yang Huiyu**

**Contact number: 18717819565**

**Email: 993492964@qq.com**

**Contents**

[1. System functions 3](#_Toc532751724)

[1.1 User module 3](#_Toc532751725)

[1.1.1 Account Management 3](#_Toc532751726)

[1.1.2 Setting personal goals 3](#_Toc532751727)

[1.1.3 Daily summary 3](#_Toc532751728)

[1.1.4 Feedbacks and suggestions 3](#_Toc532751729)

[1.2 Tomato module 3](#_Toc532751730)

[1.2.1 Before the tomato 3](#_Toc532751731)

[1.2.2 During the tomato 3](#_Toc532751732)

[1.2.3 After the tomato 4](#_Toc532751733)

[1.3 Task module 4](#_Toc532751734)

[1.3.1 Creating tasks 4](#_Toc532751735)

[1.3.2 Adding subtasks 4](#_Toc532751736)

[1.4 Analysis module 4](#_Toc532751737)

[1.4.1 Viewing historical records 4](#_Toc532751738)

[1.4.2 Analysis of works 4](#_Toc532751739)

[2. User manual 4](#_Toc532751740)

[2.1 Sample scenarios 4](#_Toc532751741)

[2.1.1 Registration and login 4](#_Toc532751742)

[2.1.2 Choosing your time pattern 4](#_Toc532751743)

[2.1.3 Starting tomato clock 5](#_Toc532751744)

[2.1.4 Finishing tomato clock 5](#_Toc532751745)

[2.1.5 Associating the tomato with tasks. 5](#_Toc532751746)

[2.1.6 Writing summary 5](#_Toc532751747)

[2.1.7 Checking for history 5](#_Toc532751748)

[2.1.8 Getting analysis 5](#_Toc532751749)

[2.1.9 Reading recommended articles 5](#_Toc532751750)

[2.2 System deployment and configurations 6](#_Toc532751751)

[2.2.1 System deployment 6](#_Toc532751752)

[2.2.2 Configurations 6](#_Toc532751753)

[3. System architecture and component design 6](#_Toc532751754)

[3.1 System architecture 6](#_Toc532751755)

[3.2 Component design 7](#_Toc532751756)

[4. Database design 7](#_Toc532751757)

[4.1 Why we chose MongoDB 7](#_Toc532751758)

[4.1.1 NoSQL 7](#_Toc532751759)

[4.1.2 MongoDB 7](#_Toc532751760)

[4.2 Concepts analysis of MongoDB 7](#_Toc532751761)

[4.2.1 Correspondence between SQL terms and MongoDB terms 7](#_Toc532751762)

[4.3 Database design 7](#_Toc532751763)

[4.3.1 Collection: Article 7](#_Toc532751764)

[4.3.2 Collection: Feedback 8](#_Toc532751765)

[4.3.3 Collection: History 8](#_Toc532751766)

[4.3.4 Collection: SequenceId 8](#_Toc532751767)

[4.3.5 Collection: Summary 8](#_Toc532751768)

[4.3.6 Collection: User 9](#_Toc532751769)

[4.3.7 Collection: Task 9](#_Toc532751770)

[4.3.8 Collection: TaskTree 9](#_Toc532751771)

[5. Other details 10](#_Toc532751772)

[Appendix A. Diagrams and pictures 10](#_Toc532751773)

# System functions

## User module

### Account Management

In the beginning of entering our program, users have to create their own accounts by filling in their username, password, email, photo, and tomato length. Then our system allocates a unique user ID for each user. After logging in, they can modify their personal information and settings. They can also logout before quitting when worry about the account security.

### Setting personal goals

Before starting to work, users can set goals for themselves. When setting, they should decide the task contents, specify the estimated time, remind time and deadline. Also, users can relate a subtask to its parent task.

### Daily summary

After working, users can do their daily summary in the program. In the summary, they can record the working efficiency of that day, like how many tasks are finished, or if there are some modifications need to be made, and everything they want to write.

### Feedbacks and suggestions

We all know that our project is not perfect. And we are always looking forward the feedbacks and suggestions of users. If they have any complaints or other related issues to express, just check out the contact information of the team.

## Tomato module

### Before the tomato

This program applies the pomodoro technique to improve the time management of users. The technique break works into small intervals (named as tomatoes, and traditionally, the length is 25 minutes) separated by small breaks. Before starting a tomato, users do some settings, like changing the tomato length.

### During the tomato

Once a tomato is started, users have to focus on their works during the period, which means they should not be allowed to interrupt the tomato in a casual way. If something urgent happens, the state of interruption will be recorded.

### After the tomato

After users finish a tomato successfully, they can associate that tomato with some tasks and then commit it. Those related data will be recorded in the database.

## Task module

### Creating tasks

Users should create their own tasks before starting to work. During the process, users need to input task name, general description, expected tomato amount, remind time and deadline.

### Adding subtasks

Maybe sometimes, users need to split their goal into several small ones. They can achieve this feature by adding subtasks to an existed task.

## Analysis module

### Viewing historical records

After working for a while, users may want to have a check of their achievements. Then, they can view historical records of that day or the entire month.

### Analysis of works

This function present users with the achievement of tasks in the form of diagrams. By analyzing their data, our program can also provide some suggestions and recommend some related articles users may be interested in.

# User manual

Our program provides the series of functions for users as below (including sample scenarios with snapshots of user interface and descriptions on system deployment and configurations):

## Sample scenarios

### 2.1.1 Registration and login

If you already have the account for Growth, you can login directly. If you haven’t, you need to register you own account by username, password, email and so on, then you can have your own account.

### 2.1.2 Choosing your time pattern

We have designed two patterns for you, adapting your own learning habits:

The first pattern allows you to set the task tomatoes. It works better when you can’t decide how much time the task will actually consume.

The second one is based on the tomato clock. That is, you can start without doing anything or writing down the daily tasks you want to complete (of course you can add tasks later). Later, when you write down tasks, you can also choose the estimate amounts of tomatoes.

### 2.1.3 Starting tomato clock

If you are ready to work for the task, please press down the start button and start your learning time. The time will be displayed on the screen in case you want to check for it. Be focus!

### 2.1.4 Finishing tomato clock

If you have some urgent things to do and you have to terminate the clock, you can press down the stop button and the clock will be stopped. But please pay attention, this tomato will be marked as uncompleted.

If you hold out till the tomato ends, you finished the tomato successfully. And congratulations, you can earn a tomato!

### 2.1.5 Associating the tomato with tasks.

In this part, you can choose from the tasks you set before, and associate the tomato with one of them. Besides, you can also build a new task to record what you have done in this tomato.

### 2.1.6 Writing summary

After a day of hard work, you may want to write a daily summary of what you have done to review and summarize. Then, you can grade your work through self-evaluation from 1 star to 5 stars.

### 2.1.7 Checking for history

You can have a check of history records to have a clear knowledge of what you have achieved recently and browse their brief descriptions. You can also choose to check the successfully-finished ones or failed ones by settings in the drop-down box.

### 2.1.8 Getting analysis

For analysis, you can check your completion status by diagram. And also, we provide you with some suggestions, like, which day in a week or which time in a day you tend to have the highest efficiency. By analysis, our team wants to help you know yourself better and choose working patterns which are more suitable for you.

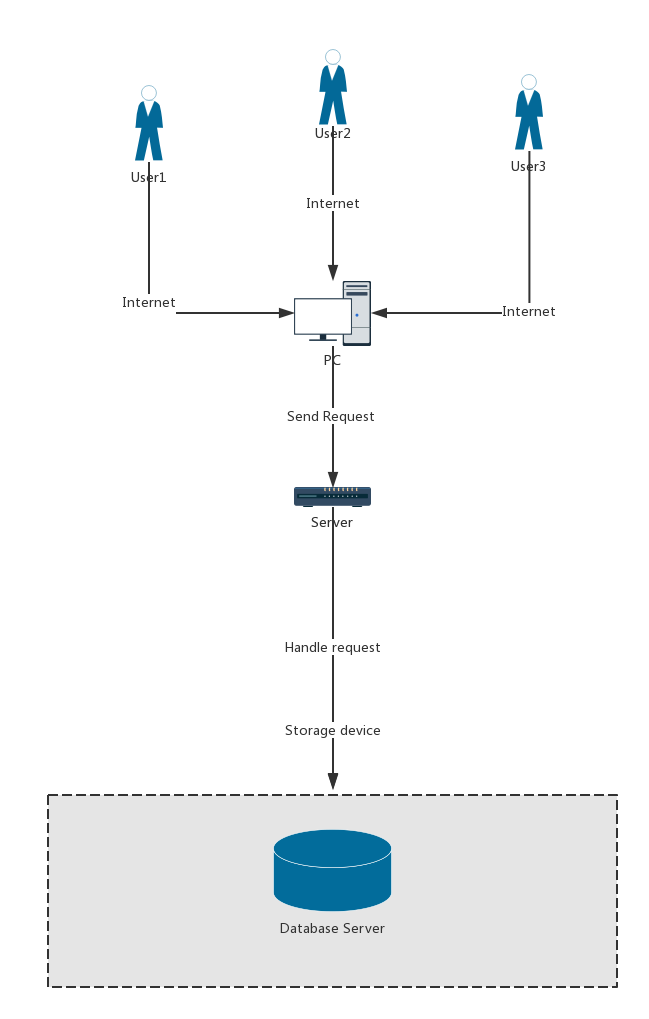
### 2.1.9 Reading recommended articles

When you want take a rest, don’t rush out of our program. We also provide you with some recommendation articles, according to your personal circumstances. You can read these articles to get knowledge and relax.

## System deployment and configurations

### System deployment

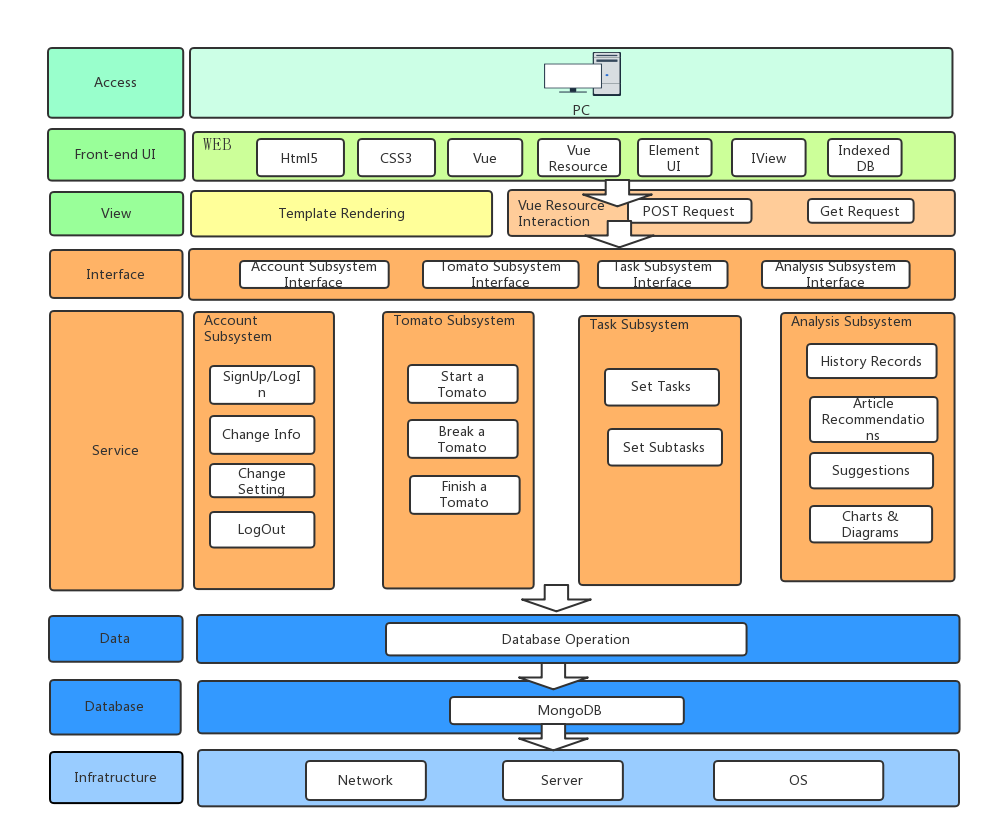
System deployment manifests the physical architecture of system software and hardware. We use deployment diagram to show it.



### Configurations

# System architecture and component design

## System architecture



## 3.2 Component design

# Database design

## Why we chose MongoDB

### 4.1.1 NoSQL

NoSQL can be used for very large scale of data storage, these types of data storage do not need a fixed pattern, no unnecessary operations can be horizontally extended. Today, we can easily access and capture data through third-party platforms (such as Google, Facebook, etc.). Users' personal information, social networks, geographical locations, user-generated data and user action logs have multiplied. If we want to mine these user data, the SQL database is no longer suitable for these applications, and the development of NoSQL database can deal with these large data very well. It has high scalability and no complex relationships. Query functionality is limited but sufficient for use in projects.

### 4.1.2 MongoDB

MongoDB is the document storage in NoSQL database. Generally, MongoDB is stored in json-like format, and the content stored is document-type. It gives you the opportunity to index certain fields and perform some of the functions of a relational database.

## Concepts analysis of MongoDB

### 4.2.1 Correspondence between SQL terms and MongoDB terms

|  |  |  |
| --- | --- | --- |
| Terms: | In SQL: | In MongoDB: |
| table | collection | Database table / collection |
| row | document | Data row / document |
| column | field | Data column / field |
| primary key | primary key | Primary key, MongoDB set “\_id” as the primary key automatically. |

## Database design

### 4.3.1 Collection: Article

| Field | Type | Description |
| --- | --- | --- |
| -id | int | Primary key |
| -tags | int | ??? |
| -title | String | The tile of the article |
| -content | String | The content of the article |
| -clickcount | String | ??? |
| -author | String | The author of the article |
| -url | String | ??? |

### 4.3.2 Collection: Feedback

| Field | Type | Description |
| --- | --- | --- |
| -id | long | The primary key |
| -time | Date | The time when user committed the feedback |
| -userid | long | The id of the user who committed the feedback |
| -content | String | The content of the feedback |
| -answer | String | The process of the feedback |

### 4.3.3 Collection: History

| Field | Type | Description |
| --- | --- | --- |
| -id | long | Primary key |
| -userId | long | The id of the user |
| -startTime | Date | The time when user start the tomato |
| -endTime | Date | The time when user end the tomato |
| -tomatoLength | int | The length of the tomato |
| -status | int | The status of the tomato |
| -taskId | long | The id of the task which is associated with the tomato |

### 4.3.4 Collection: SequenceId

| Field | Type | Description |
| --- | --- | --- |
| -id | String | Primary key |
| -sequenceId | long | The current id of the collection |
| -collName | String | The name of the collection |

### 4.3.5 Collection: Summary

| Field | Type | Description |
| --- | --- | --- |
| -id | long | Primary key |
| -userId | long | The id of the user |
| -content | String | The content of the Summary |
| -time | Date | The time when user wrote the summary |
| -selfRating | int | The score which users give themselves |

### 4.3.6 Collection: User

| Field | Type | Description |
| --- | --- | --- |
| -id | long | Primary key |
| -username | String | The name of the user |
| -password | String | The password of the user |
| -tomatoLength | int | The amount of the tasks user has finished |
| -tomatoWeekly | int | The amount of the tasks user has finished this weekly |
| -email | String | The email of the user |
| -userFace | String | The userface of the user |

### 4.3.7 Collection: Task

| Field | Type | Description |
| --- | --- | --- |
| -id | long | Primary key |
| -name | String | The name of the task |
| -description | String | The description of the task |
| -expectedTomato | int | The amount of the tomato that user expect to use to accomplish the task |
| -tomatoCompleted | int | The amount of the tomato that user has accomplished with the task |
| -setTIme | Date | The time when user set the task |
| -deadline | Date | The time when user should accomplished the task |
| -remindTime | Date | The time when user want to be reminded to start the task |
| -finishedTime | Date | The time when user finished the task |
| -status | int | The state of completion of a task |

### 4.3.8 Collection: TaskTree

| Field | Type | Description |
| --- | --- | --- |
| -sonTaskId | long | Primary key |
| -parentTaskId | long | The id of the parenttask |
| -sonTaskName | String | The name of the sontask |
| -isTaskDone | boolean | Whether the sontask has been done. |

# Other details

## 5.1 Other techniques

We use fudannlp, which is a natural language processing package, to analyze users’ tasks and summaries to get the keywords of users. This technology omits the common words in the text and gets the keywords list according to the rare degree. In this way, we can get the keywords in the text. However, I think the accuracy of this method is not high enough and needs to be improved.

# Appendix A. Diagrams and pictures