6 things must to do

Break manerism by ivy lee method

Changhoi Kim
Dept. of Information System
Hanyang Univ
Seoul, Korea
changhoi0522@hanyang.ac.kr

Jungeun Kim
Dept. of Information System
Hanyang Univ
Seoul, Korea
wjddms04@hanyang.ac.kr

Eunchai Kim
Dept. of Information System
Hanyang Univ
Seoul, Korea
eunchai512@naver.com

TABLE I
ROLE ASSIGNMENTS

Role	***Name***	***Task***
User	NUGU Speaker and Application user	Application can use for free on IOS and AOS. Every NUGU Speaker and application users can use default feature of application.
Customer	SKT and Application User	Customer want to solve the problems at an acceptable cost . SKT want application which is running on NUGU speaker, so more NUGU users get advantages. Application users who want more advanced feature of application will pay subscription fee.
Software Developer	Changhoi Kim, Jungeun Kim	Engineers create actual product. They demand core function requirements, which give then up to their developing process, to manager. Software developers' goal is to satisfy users and customers by realizing close to the requirements.
Development Manager	Eunchai Kim	Development managers intercommunicate with Customer and Software developer to satisfy with whole stakeholders. They schedule timeline to stick to the project before deadline. They also build overall developing process and support software developers.

***Abstract*—"6 things must to do" is an application that helps people practice Ivy Lee Method, a way to get rid of mannerisms and manage time efficiently. We use artificial intelligence speaker NUGU to help users set up 6 things each day and manage tasks to be done every day**

***Index Terms*—ivy lee method, 6 things, mannerism, nugu, ai speaker, todo list**

I. INTRODUCTION

Ivy Lee Method is a technology for efficient time management used over a long period of time. It is proposed by Ivy Lee to executives and employees to eliminate corporate manners and improve productivity. Next are the following the five steps:

1) Set up six things to do for next day before bed.

2) Arrange in the order of priority among the six things.

3) The things are executed according to the priority set, and the next thing cannot be carried out until the previous thing is completed.

4) If the assignment is not completed, the remaining tasks will be carried out the next day.

5) Repeat the above method every day.

This method makes modern people, surrounded with many tasks, think of the most important ones to be carried out. Since the method is simple enough for anybody to follow, it can be implemented until the execution stage. Also to eliminate the fear of starting from scratch. Furthermore, this helps break down mannerism and create a productive day.

The most complete application derived from the existing Ivy Lee Method was the application called "6 things". This application is a To-Do list app that writes down six things to do. However, it does not force a priority, but rather only writes down six To-Do list. Also it does not store the degree of achievement of other days. Moreover, record cannot be checked, feedbacks cannot be made and no retrospection.

"6 things must to do" sends an alarm so that people can set the day or the next day task. Also sets the To-Do list and checks the list of achievement through the AI speaker NUGU. Using NUGU, people can also check the next thing to be completed or task that was completed. Through subscription services, we provide a dashboard to store daily records and check weekly and monthly attainment rates.

II. REQUIREMENTS

A. Mobile Application

Mobile applications will be used by users. Both Android and iOS are available. Cross Platform applications to

Identify applicable funding agency here. If none, delete this.

communicate with the server is required. There will be five four functions(login, task list, dashboard, social) in the application.

1) Login: Users only use social login. So there are no local application login form. And there are no signup screen because the application will use the information which is from social login. There will be only simple login screen.
   a) Additional information screen: Social login might give email and unique app id value. But if user refuses to provide an email and the user is new client, the user should type email manually for using application.

2) Navigation tab: The Application has some features, so it need the navigation bar for split the domain. It is on the bottom of the screen.

3) Tasks (Main Screen): Six tasks screen will be on the main screen. In addition to the task designation, various works will be available.
   a) One task component: There is task component which shows the detail of one task.
      i) Read tasks detail: By default, the component shows the detail of one task. it has ”when”, ”where”, ”with whom” fields, and simple to-do list to complete the task.
      ii) Set details of task: In this component, user can make detail of the task. The detail fields are as mentioned above.
      iii) Update to-do status in a task: Each to-do in the to-do list of a task can be completed. When all the to-dos are completed, the task is completed, too. However if there are no detail to-do, user can complete the task by other button in detail component or in list component.

   b) Task list component: There is task list component which has plural task components.
      i) Read task list of specific date: In this screen, the component shows the list of tasks. It is default screen of task list component.
      ii) Add task: In task list screen, user can add task for next day (or next session of task list). The added task is consist of task component.
      iii) Start tasks and lock the list: When user click lock button, the list become uneditable except complete the task. Lock time is set by user in setting screen.
      iv) Check ongoing task and complete the task: When the task list is locked, user can find what task should be handled, and check the task as complete. If the task has specific to-do list, and user check the task as complete, all to-dos in the task is also checked as complete.
      v) Pass the remain incomplete tasks to next task list: When the session (user-set lock time) is over, and the list has remain tasks, The remaining list goes into the next (session's) task list.

4) Setting: User can set the option of the application. They can adjust alarm, Task list session time (Lock time), and change their account information in this component.

5) Dashboard: User can check the usage of the application in this component. There are graph for visualizing monthly, weekly, daily task processing degree. And also check the list of the specific date.

B. NUGU Service

Also, user can access the application through NUGU AI speaker. User speak specific command and the NUGU speaker response and process the command.

1) Check ongoing task: User can complete the task which is currently in progress.

2) Complete the task: User can complete ongoing task

3) Check the remain tasks: User can check the remain tasks in the list.

III. Development Environment

A. Choice of software development platform

1) Which platform and why
   Our service operates mainly on mobile applications. We use Cross Platform application that is available for both Android and iOS operating systems and we develop app focusing on the mobile OS latest version due to development environmental limitations (React Native limitations) and time constraints. We are using React Native since it can develop both platforms. The reason why we choice mobile application based, not the web, is that it needs to have easy access to the tasks and provide alarm service to the specific ones.

   The server application will work on work on Linux based Lambda of AWS. And use AWS DynamoDB as database. Lambda and DynamoDB can save the cost when build simple applications because of its generous free tier.

   And we do not have to worry about infrastructure and can focus on server application logic when we use managed services of AWS like DynamoDB and

Lambda. In addition, we thought that golang is a suitable language to operate in Lambda system since it remains simple executable binary after compilation and does not have to run virtual machine like JVM.

2) Which programming language and why
   a) Typescript
      Typescript, Javascript's superset, provides more stable development environment. By defining the interface and types, errors caused by misusing the type are prevented in advance. It can also easily identify the specific way of using methods and functions. In addition to the above advantages, it is freely available to use the large opensource system of JavaScript.

   b) Go
      Go language is a compile language, that has fast performance as C language. There is no complicated grammar, and by unifying some of the code styles, it reduces the concern about the convention. By consuming computing power efficiently, relatively little computing resources are required to achieve the same goal as other languages.

   c) Finally, both languages are growing rapidly in terms of language utilization, especially typescript is becoming the standard for JS development environments. Golang also has a huge growing community and is in line with recent technology trends.

3) Provide a cost estimation for your built

TABLE II
COST ESTIMATION

Name of software / hardware	***Cost***
Apple Developer Program	129,000 won
AWS Lambda	0 dollar
AWS DynamoDB	0 - 5 dollar

4) Provide clear information of your development environment
   - MacOS Catalina 10.15.7
     - 2.3 GHz 8 Core intel Core i9
     - 32GB 2667 MHz DDR4 Memory
   - Windows 10
     - 1.80 GHz 8 Core Intel i7
     - 8GB Memory
   - Visual Studio Code 1.50.1
   - Goland 11.0.8 (2020.2)
   - Go 1.15
   - Gin 1.6.3
   - Node 12.18.3
   - Typescript 4.0.3
   - React Native 0.63.3
   - Git 2.28.0
   - Other open sources
     - **Link: Open source used in mobile application**
     - **Link: Open source used in the server**

5) Using any commercial cloud platform

   AWS Lambda and DynamoDB will be used to launch production servers. Computing and Database resources are same the information written above, and AWS API Gateway service will be used to proxy HTTP requests to Lambda.

B. Software in use

1) Git
   Git is a distributed version-control system for tracking changes in source code during software development. It manage the code version and make co-working with other programmer more easier. There is branch system which is core of co-working in git-managed source code with many engineer. Each developer write code for different features and then, the codes can be merged easily by git.

2) Github
   GitHub is an American multinational corporation that provides hosting for software development and version control using Git. It has features about git and also has own feature about source access control. Source code can be managed by permission in github.

3) Golang and Gin
   Golang is a statically typed, compiled programming language designed at Google. Golang is known as fast as C and maintainable as JAVA. It is compilable language and when it is compiled, a single binary file is created. The binary file can excutable in various platform (Go build command has OS option which is about what platform the binary will be excuted in).
   Gin is go web framework. It's full name is gin-gonic. It helps engineer to build web application server easily. It is opensource and maintained by many contributers.

4) AWS
   AWS stands for amazon web service, which provides cloud computer. AWS has many availability zone, so engineer can build safe and solid application. AWS provides various computing type, and there are many managed services. Managed service means that engineers do not have to concern about infrastructure. AWS ensures that the service logic will work. That services like Lambda, DynamoDB, etc is called AWS

serverless services that make developer to focus to build application, not infratructure.

5) Typescript
Typescript is superset of javascript. Javascript is not type-safety language, so using javascript in huge project can make dangerous bug. Typescript can be used in javascript project, and also only typescript project. It is opensource and has huge engineer commnunity and maintained by Microsoft.

6) React Native
React Native is framework for mobile cross platform. This framework is created by Facebook. It works on Typescript and Javascript environment. TS/JS code create Native code which is written in JAVA (Android) and Objective-C(iOS)

C. Task distribution

TABLE III
TASK DISTRIBUTION

Name	***Task Distribution***
Changhoi Kim	Build servers, including databases, and manage the whole project (Server, DB)
Jungeun Kim	Connect the server and the mobile app, and the data values received from the server (Redux, Container)
Eunchai Kim	Align the received data and design UI and CSS (Screen)

Our roles were distributed by technology rather than by app function. Our app has multiple functions, and we tried to divide the roles by the functions (ex. Making tasks/ Managing tasks/ Adding friends). However, we found out this method is inefficient, and it would be difficult to complete the project if we proceed it in this way. It is because we realized that it is almost impossible to learn all the different skills to create a single function in time. Therefore, we end it up decided to divide the tasks by component of the app rather than the function, and let each person take charge of it.

IV. SPECIFICATIONS

A. Mobile Application

1) Login: Our application has only Google login. There are no other process for sign up or sign in. Social login provide email, unique app Id, etc. Using this information, application get token from api server, and save it in the device local storage. In server, take social account information and get user data in DynamoDB or create user, and then create JWT which is used when user access api service.

```
# in Application
socialAccount := googlaLogin()
token := getToken(socialAccount)
saveTokenInLocalStorage(token)
```

```
# in API Server
socialData := getRequest()
user := getUserOrCreate(socialData)
token := issueToken(user)
response(token)
```

2) Navigation tab:
Navigation tab on the bottom of screen is provided by opensource named React Navigation. And there is no other Network I/O

```
onChangeTab(targetScreen) {
    currentScreen = targetScreen
    re-render()
}
```

3) Tasks (Main Screen): In main screen, authentication is required. Token should be stored in local storage and used when call API server service.
   a) One task component: In task component, there are details of specific task. For showing the detail, API call is required. And in application, there are global store for saving data state. All data is stored in one global redux store, and it is only updated by dispatch function.

```
# in Application
data := apiCall(type, taskID)
# CRUD API call
dispatch(updateStore(data))
    .then(re-render())
```

```
# in API server
type, taskID := request
taskTable(where id = taskID)
    .action(type)
# CRUD of task detail
```

   b) Task list component:
   Task list component data are also managed by Redux store. All API call and result update the store through redux's dispatch function like above.

4) Setting:
In setting screen, there are features about alarm, updating user information, setting task session time. All but updating user information use local device's options. Alarm and task session time use device's timer API. And updating user information use API call.

```
# lock session time
default = 8 * HOUR
lockSessionTime = default
onUpdateSessionTime((time) => {
    lockSessionTime = time
})

# alarm
```

```
default = 3 * HOUR
triggerTimer = default
onUpdateTriggerTime((time) => {
    triggerTimer = time
})

# update user info
apiCall(updateData)
```

5) Dashboard:
   In this component, the application should be able to get the log of user's usage by period. Data is stored in DynamoDB, so API call is required. And also the data is managed by redux.

```
# in Application
data := getLogOf["weekly"]()
dispatch(updateDashboard(data))
    .then(re-render())
```

B. NUGU Service

Every utterance command is sended to processing backend server through NUGU SDK. NUGU SDK process the voice to string. And Backend proxy server change the string to valid command form and send it to the api server.

```
voice > NUGU SDK > Processed voice string
> Backend Proxy server > API Server
```