Internship Report - YanTuTech

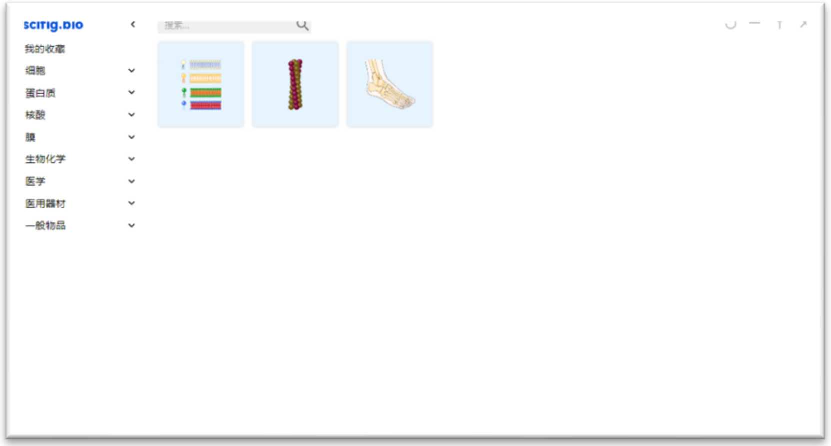
Yufeng Xiao

Internship Overview

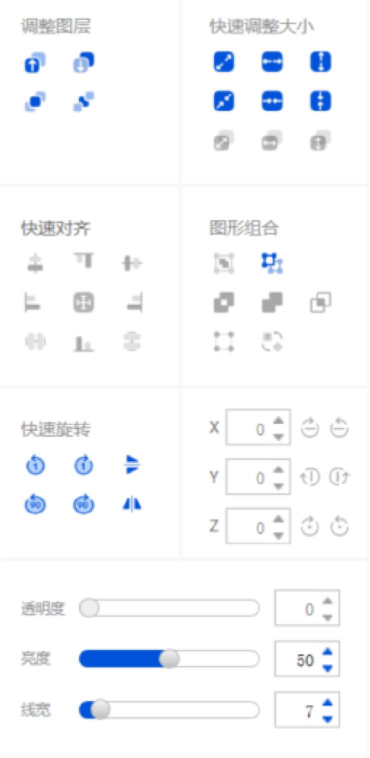
Company:	YanTuTech（研涂科技）
Location:	Zurich
Format:	Remote
Supervisor:	Zezhe Huang
Coordination:	GitHub teambition zoom
Duration:	27.06 - 18.09

YanTuTech – A Startup from Shanghai

A startup founded in 2020 in Shanghai, committed to make **creation** of **scientific figures** easier for researchers with the help of the **PowerPoint add-in “scifig”**.

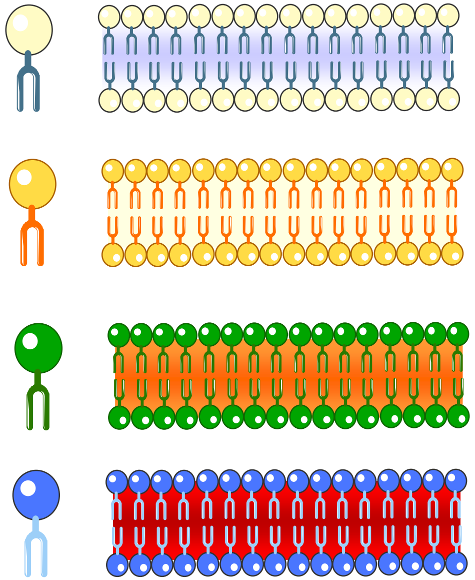


Library of scientific figures

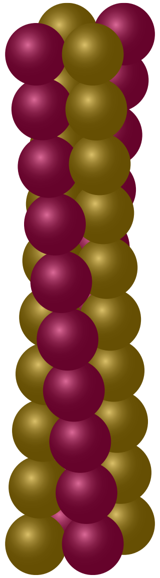


Side bar tools

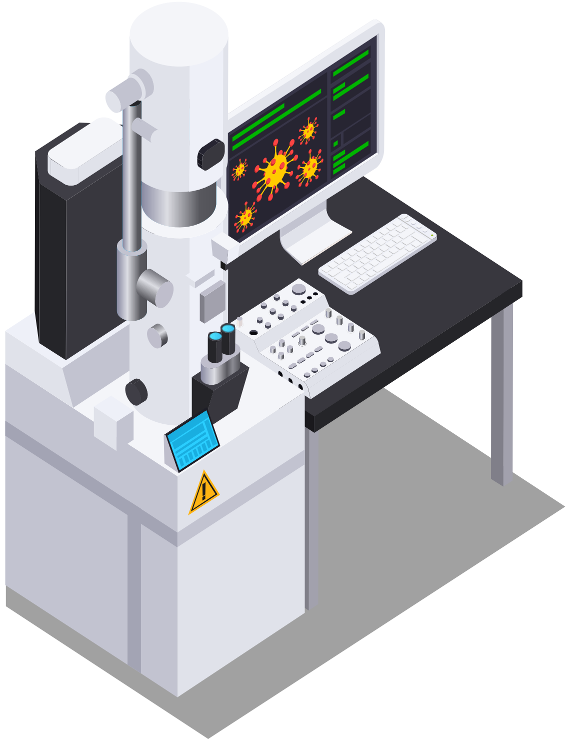
Some Scientific Figures



Cell membrane



Nucleic acid

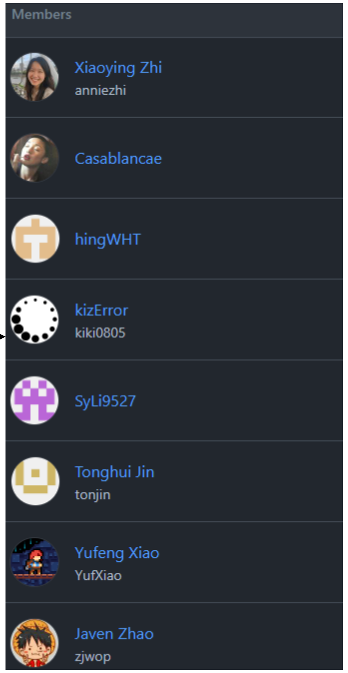


Lab equipment

Members – A Distributed Team

Members are working **part-time** or full-time on this project, from Shanghai, Stockholm and Zurich

Development team



Members

Xiaoying Zhi
anniezhi

Casablancae

hingWHT

kizError
kiki0805

SyLi9527

Tonghui Jin
tonjin

Yufeng Xiao
YufXiao

Javen Zhao
zjwop

ETH

UZH

KTH VETENSKAP OCH KONST

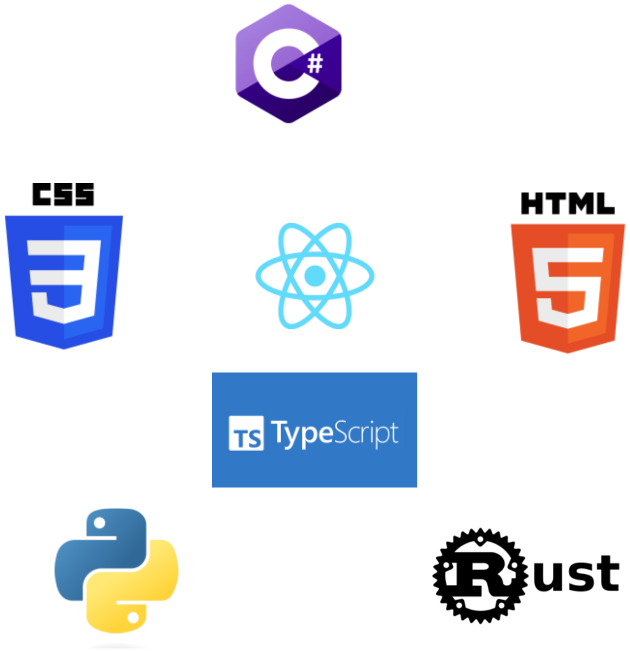
netlight

Ctrip

ByteDance

Technical Stack

Programming language



Database Management

PostgreSQL

Task Scheduling & Message Broker

CELERY
RabbitMQ

Technical Stack

Memory-caching System

Memcached

Frameworks

django

Microsoft .NET WPF

Microsoft Visual Studio Tools for the Microsoft Office System

VSTO

ELECTRON

Tauri

Web Server in Production

NGINX

Technical Stack

Monitor system

Prometheus

Grafana

Development Environment

Windows

docker

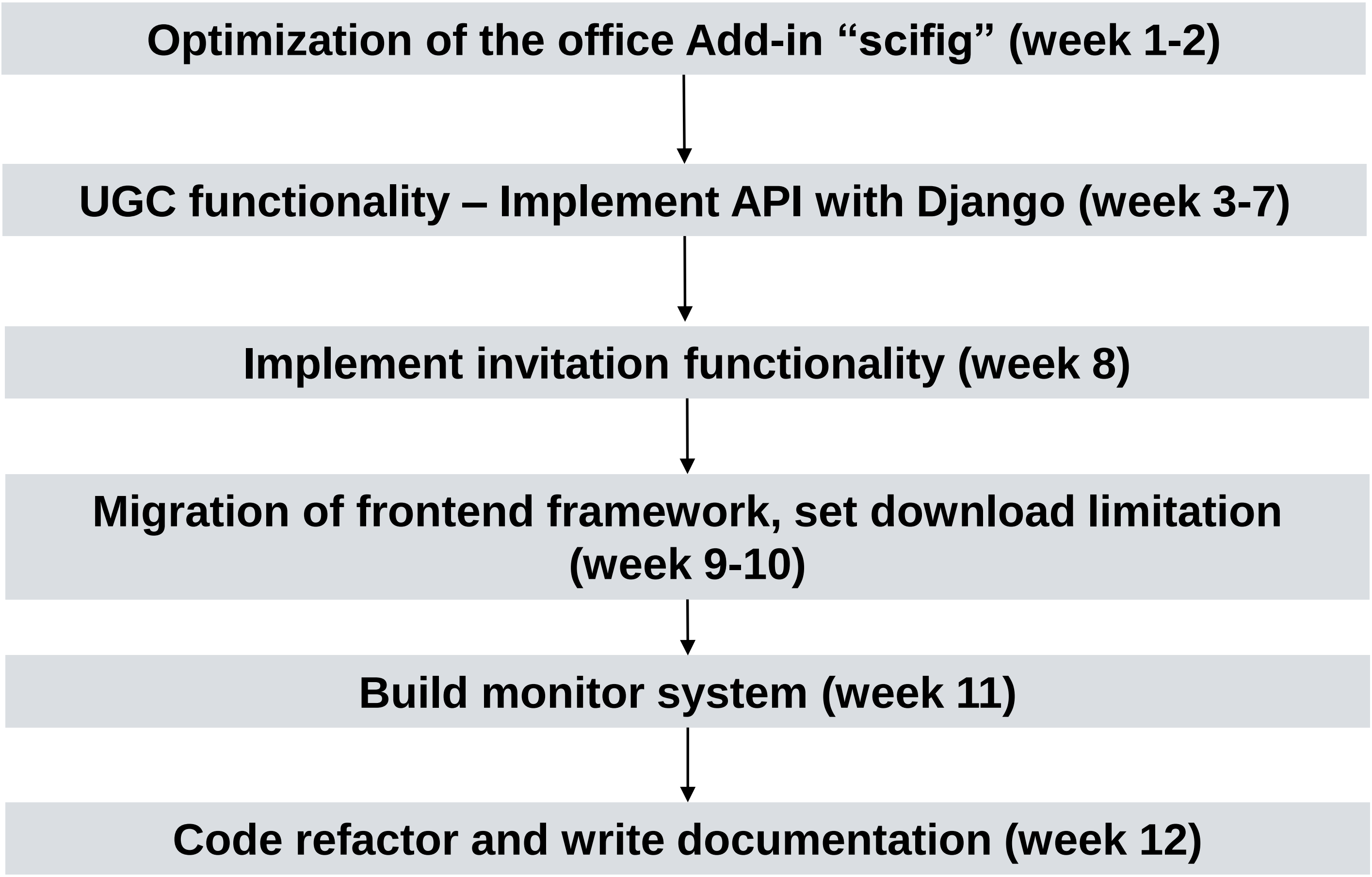
Windows Subsystem for LINUX

Cloud Service

Tencent Cloud

serverless

Tasks During Internship



Optimization of the office Add-in “scifig” (week 1-2)

UGC functionality – Implement API with Django (week 3-7)

Implement invitation functionality (week 8)

Migration of frontend framework, set download limitation (week 9-10)

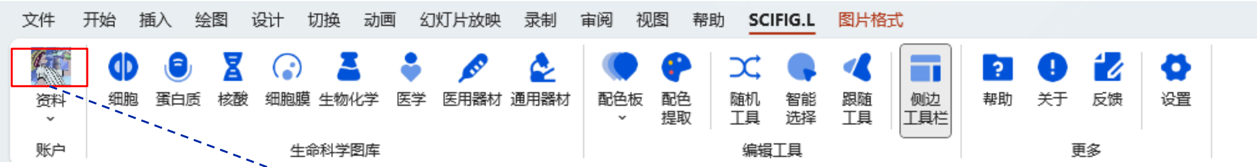
Build monitor system (week 11)

Code refactor and write documentation (week 12)

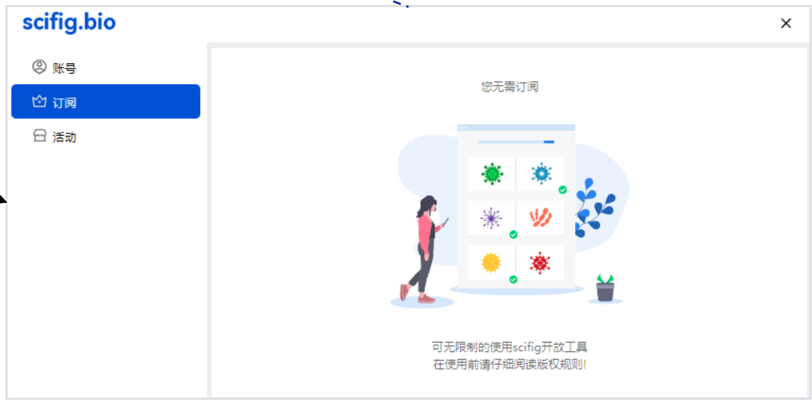
Optimization of the office Add-in “scifig” (week 1-2)

Optimize UI

Microsoft® .NET WPF



ELECTRON



您无需订阅

可无限制的使用scifig开放工具
在使用前请仔细阅读版权规则!

Fix bugs

Microsoft® Visual Studio
Tools for the Microsoft Office System

Functionality: e.g. change transparency of shapes

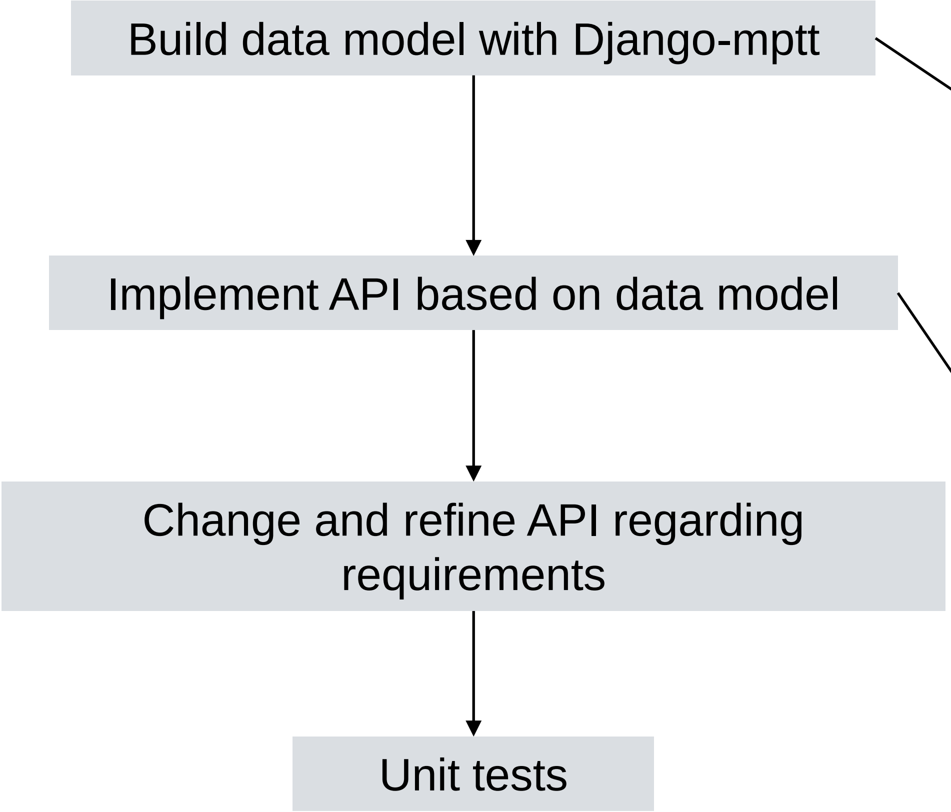
UGC functionality – Implement API with Django (week 3-7)

For the **user generated content**, we need to create a **storage space** for users, where user can store figures and share it directly in this add-in.

Implemented **Shared-Space** which is functionally similar to **Google Drive**.

django
PostgreSQL
Tencent Cloud
RabbitMQ
CELERY

Shared Space – A Tree-Structured Space for Storage

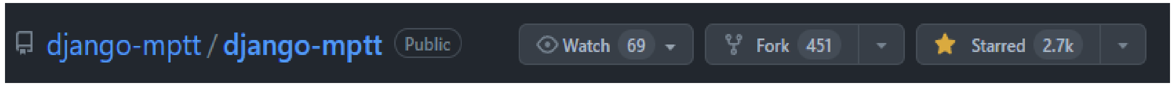


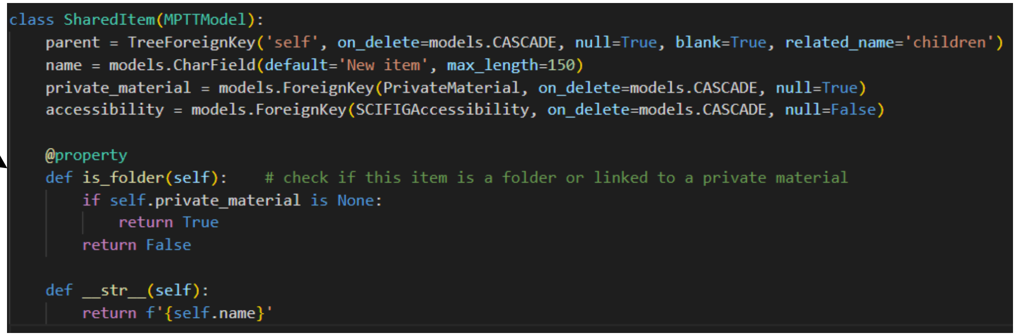
Build data model with Django-mptt

Implement API based on data model

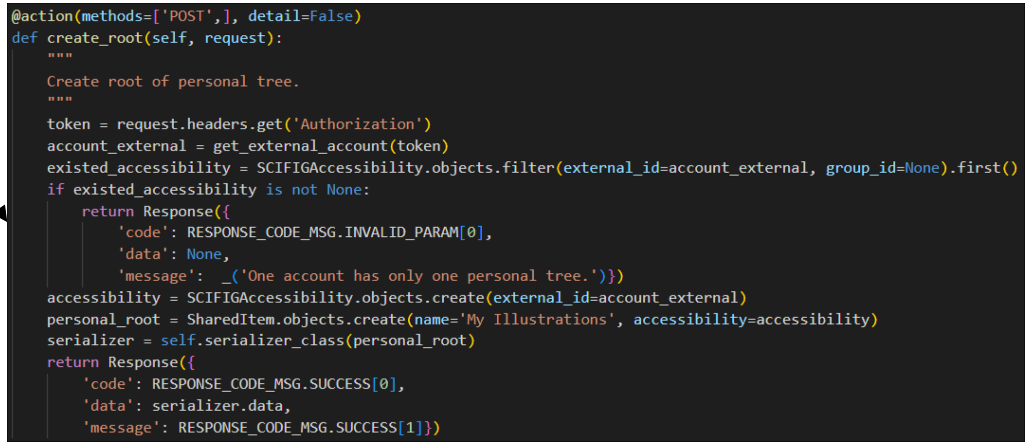
Change and refine API regarding requirements

Unit tests



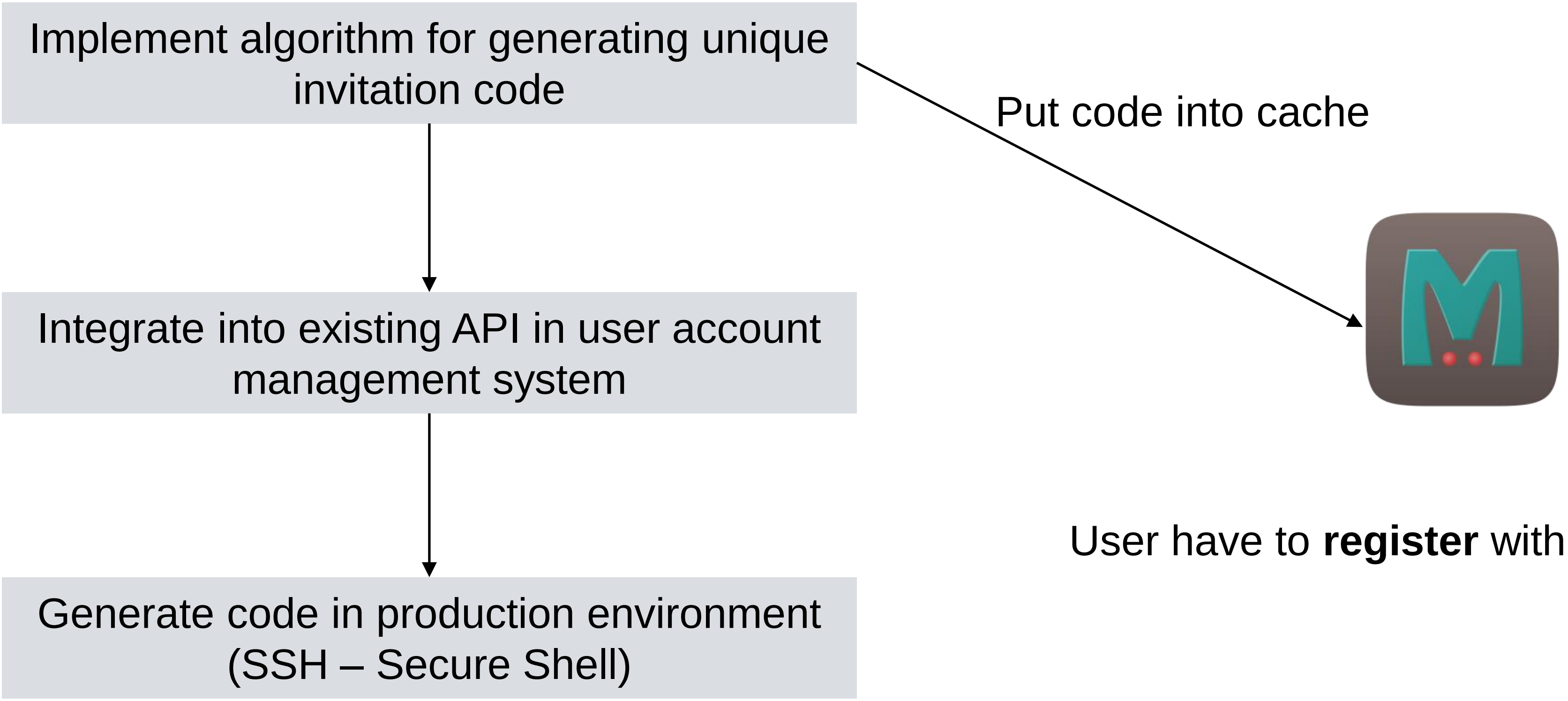


```python
class SharedItem(MPTTModel):
    parent = TreeForeignKey('self', on_delete=models.CASCADE, null=True, blank=True, related_name='children')
    name = models.CharField(default='New item', max_length=150)
    private_material = models.ForeignKey(PrivateMaterial, on_delete=models.CASCADE, null=True)
    accessibility = models.ForeignKey(SCIFIGAccessibility, on_delete=models.CASCADE, null=False)

    @property
    def is_folder(self):    # check if this item is a folder or linked to a private material
        if self.private_material is None:
            return True
        return False

    def __str__(self):
        return f'{self.name}'
```



```python
@action(methods=['POST',], detail=False)
def create_root(self, request):
    """
    Create root of personal tree.
    """
    token = request.headers.get('Authorization')
    account_external = get_external_account(token)
    existed_accessibility = SCIFIGAccessibility.objects.filter(external_id=account_external, group_id=None).first()
    if existed_accessibility is not None:
        return Response({
            'code': RESPONSE_CODE_MSG.INVALID_PARAM[0],
            'data': None,
            'message':  _('One account has only one personal tree.')})
    accessibility = SCIFIGAccessibility.objects.create(external_id=account_external)
    personal_root = SharedItem.objects.create(name='My Illustrations', accessibility=accessibility)
    serializer = self.serializer_class(personal_root)
    return Response({
        'code': RESPONSE_CODE_MSG.SUCCESS[0],
        'data': serializer.data,
        'message': RESPONSE_CODE_MSG.SUCCESS[1]})
```

Implement invitation functionality (week 8)



Implement algorithm for generating unique invitation code

Integrate into existing API in user account management system

Generate code in production environment (SSH – Secure Shell)

Put code into cache

User have to **register** with an invitation code

Migration of frontend framework, set download limitation (week 9-10)

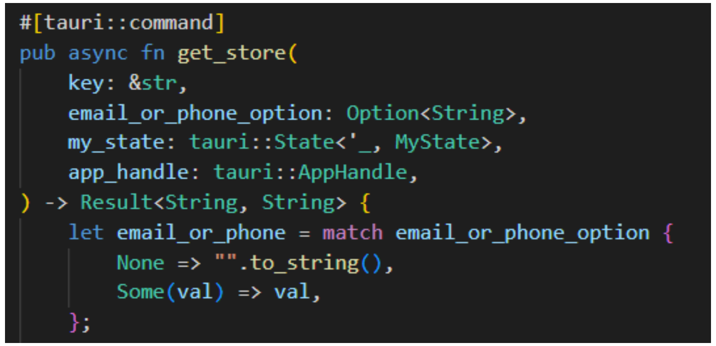
Electron and tauri are used for **inter-process communication**

Migrate frontend framework from electron to tauri

↓

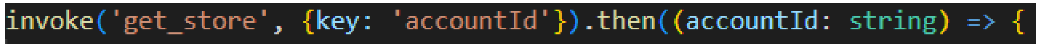
Implement commands in Tauri to limit download of figures

Tauri command

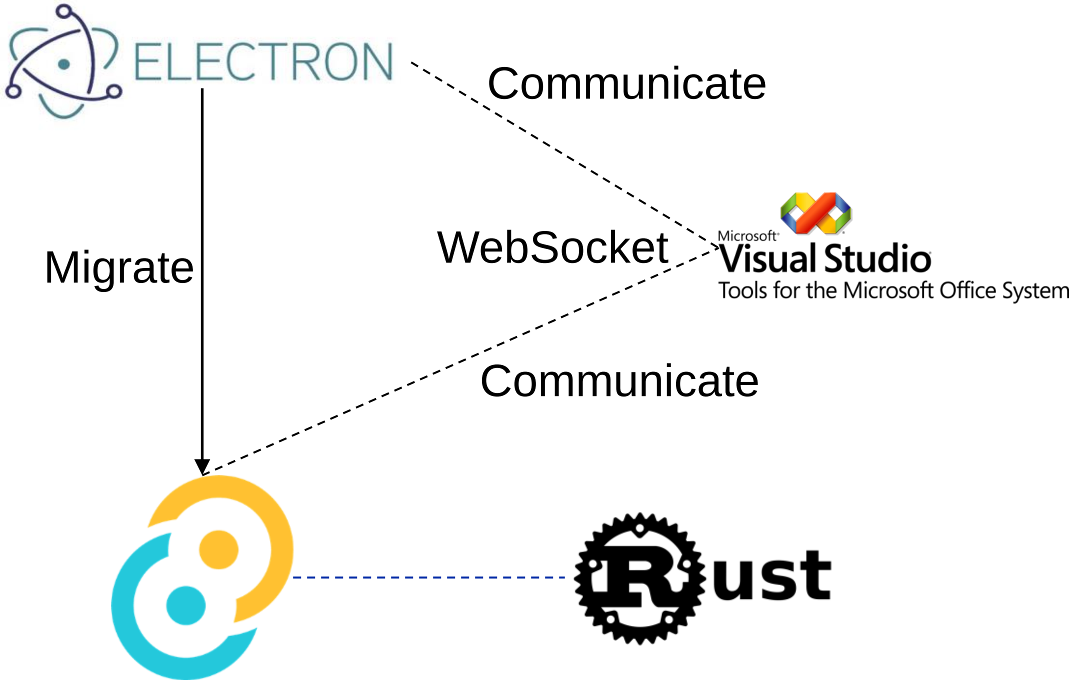


```
#[tauri::command]
pub async fn get_store(
    key: &str,
    email_or_phone_option: Option<String>,
    my_state: tauri::State<'_, MyState>,
    app_handle: tauri::AppHandle,
) -> Result<String, String> {
    let email_or_phone = match email_or_phone_option {
        None => "".to_string(),
        Some(val) => val,
    };
```

Invoke in Typescript



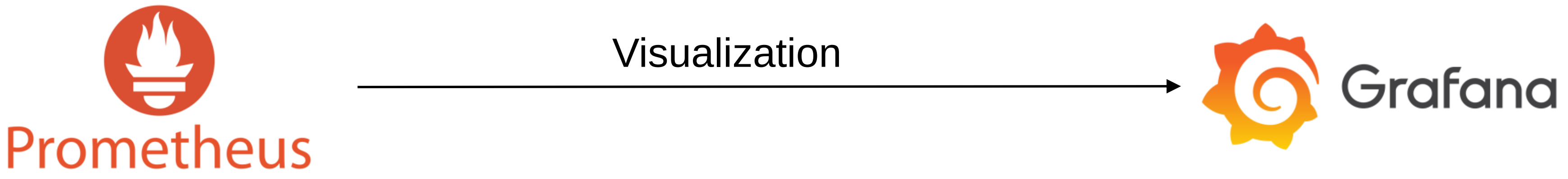
```
invoke('get_store', {key: 'accountId'}).then((accountId: string) => {
```



Tauri

More stable and less bugs

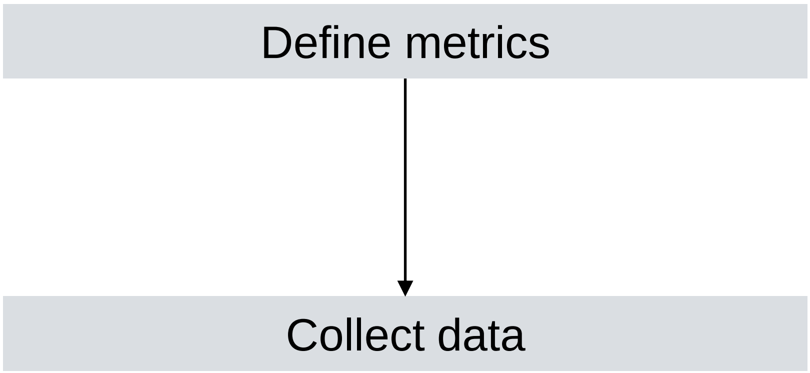
Build monitor system (week 11)



Prometheus

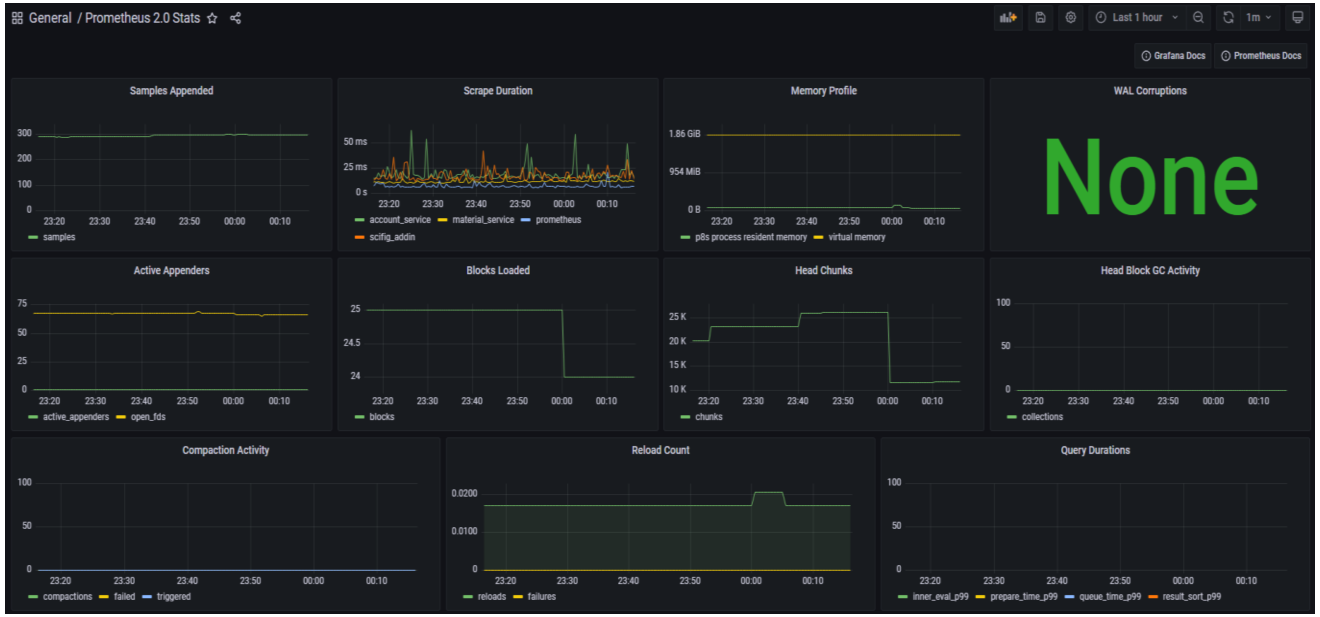
Visualization

Grafana

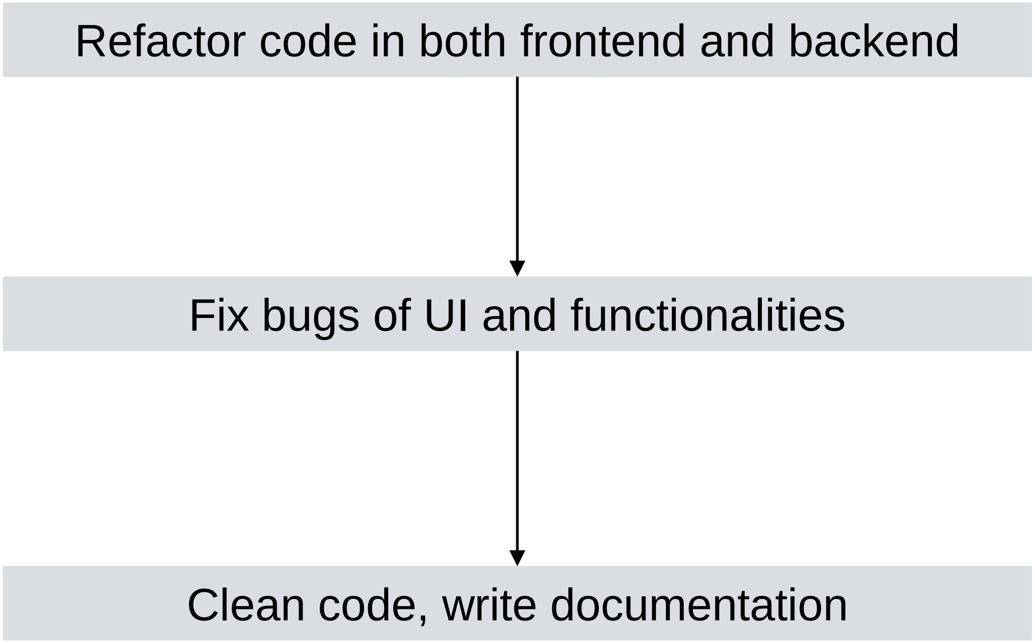


Define metrics

Collect data



Code refactor and write documentation (week 12)



Refactor code in both frontend and backend

Fix bugs of UI and functionalities

Clean code, write documentation

Summary

Gained a good insight into **professional** workflow,

Understanding in different **frequently** used technologies in **industry**,

Remote work **not** always efficient,

Overall a very **meaningful** internship.